Die approbierte Originalversion dieser Diplom-/Masterarbeit ist in der Hauptbibliothek der Technischen Universität Wien aufgestellt und zugänglich.

http://www.ub.tuwien.ac.at

The approved original version of this diploma or master thesis is available at the main library of the Vienna University of Technology.

http://www.ub.tuwien.ac.at/eng

FAKULTÄT FÜR !NFORMATIK

Faculty of Informatics

Kontextsensitive Planung von Langzeitarchivierungsmaßnahmen

DIPLOMARBEIT

zur Erlangung des akademischen Grades

Diplom-Ingenieur

im Rahmen des Studiums

Software Engineering and Internet Computing

eingereicht von

Michael Kraxner
Matrikelnummer 9925916

an der Fakultät für Informatik
der Technischen Universität Wien

Betreuung: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber
Mitwirkung: Dr.techn.Dipl.-Ing. Christoph Becker

Wien, 1. März 2015

Michael Kraxner | Andreas Rauber

FAKULTÄT
FÜR !NFORMATIK

Faculty of Informatics

Context Aware Preservation Planning

DIPLOMA THESIS

submitted in partial fulfillment of the requirements for the degree of

Diplom-Ingenieur

in

Software Engineering and Internet Computing

by

Michael Kraxner
Registration Number 9925916

to the Faculty of Informatics
at the Vienna University of Technology

Advisor: Ao.Univ.Prof. Dipl.-Ing. Dr.techn. Andreas Rauber
Assistance: Dr.techn.Dipl.-Ing. Christoph Becker

Vienna, 1st March, 2015

Michael Kraxner

Andreas Rauber

Erklärung zur Verfassung der Arbeit

Michael Kraxner
Guneschgasse 4, 1190 Wien

Hiermit erkläre ich, dass ich diese Arbeit selbständig verfasst habe, dass ich die verwendeten Quellen und Hilfsmittel vollständig angegeben habe und dass ich die Stellen der Arbeit – einschließlich Tabellen, Karten und Abbildungen –, die anderen Werken oder dem Internet im Wortlaut oder dem Sinn nach entnommen sind, auf jeden Fall unter Angabe der Quelle als Entlehnung kenntlich gemacht habe.

Wien, 1. März 2015

Michael Kraxner

Acknowledgements

Part of the following work was supported by the European Union in the 7th Framework Program, IST, through the SCAPE project, Contract 270137.

I want to thank my supervisor, Prof. Andreas Rauber, for his constructive and helpful comments, and the opportunity to work in the SCAPE project and as part of his team.

I also want to thank my assistant supervisor, Dr. Christoph Becker, for his suggestions and guidance for this thesis, and support for the related work within this project that not always was easy.

Thanks to my project partners and my colleagues for all the discussions, their knowledge, and their complementary work which made this thesis possible.

I wish to thank my family and friends, and especially my parents for their long lasting support and for keeping faith in me after all this time. And my family in law for their help which allowed me to finish this thesis.

Danke Klara, weil die Welt mit dir ein noch wunderbarerer Ort ist.

And I want to thank Aitziber, for just being there, assisting me in my nine o'clock crises and dispelling my doubts, and bringing easiness to my life with her humor and spirit.

Kurzfassung

Der Umfang an digitalen Inhalten nimmt rasant zu und stellt Organisationen mit der Verpflichtung diese auf lange Sicht zu erhalten vor neue Herausforderungen. Die Planung von Langzeitarchivierungsmaßnahmen zielt darauf ab, die beste Strategie zu ermitteln um den Zugriff auf diese Inhalte für eine spezifische Zielgruppe über einen längeren Zeitraum zu ermöglichen. Die Entscheidungsfindung sollte dabei durch einen wiederholbaren Prozess auf Basis von objektiven Kriterien erfolgen, und zu reproduzierbaren Ergebnissen führen, die das Vertrauen in die Entscheidung stärken. In den letzten Jahren waren zahlreiche Initiativen darum bemüht, damit verbundene Probleme auf verschiedenen Ebenen zu lösen, und haben zu neuen Erkenntnissen bezüglich der Herausforderungen für die Erhaltung von digitalen Daten geführt. Insbesondere wurde ein Arbeitsablauf für die Planung von Langzeitarchivierungsmaßnahmen definiert. Dieser bietet eine Anleitung, um alle relevanten Fakten - über die Organisation, die Inhalte, Anforderungen, und potentielle Strategien - zu sammeln. Experimente werden mit einer repräsentativen Auswahl von Objekten durchgeführt, und die Ergebnisse in Hinblick auf definierte Anforderungen evaluiert. Sie bilden die Basis um eine informierte Entscheidung zu treffen, und resultieren in einen ausführlich dokumentierten Plan.

Bisher ist die Unterstützung durch Tools beschränkt, und auf Grund mangelnder Automatisierung ist es noch immer eine hauptsächlich manuelle und aufwendige Aufgabe. Informationen über Inhalte und die Umgebung müssen manuell gesammelt werden, und Anforderungen werden meist ad-hoc oder Tool-spezifisch beschrieben, was deren Wiederverwendung sowie den Einsatz für andere Zwecke erschwert. Prototypen für Register von Strategien und Evaluierungstools sind zum Teil integriert, jedoch sind diese schwer wart- und erweiterbar. Ebenso können resultierende Pläne im Allgemeinen nicht automatisch ausgeführt werden.

Diese Arbeit zielt darauf ab, den Planungsprozess durch Integration mit dem Kontext zu verbessern und den Grad der Automatisierung zu steigern. Dazu werden vorhandene Informationsquellen und Systeme identifiziert, und eine Ontologie wird zur Beschreibung von zusammenhängenden Konzepten verwendet. Um die Integration zu vervollständigen werden fehlende Schnittstellen definiert, und es wird eine Methodologie zur Spezifikation von Bedingungen zur Qualitätskontrolle eingeführt. Eine Prototypimplementierung des Planungstools wird erweitert und anhand einer Fallstudie evaluiert. Es zeigt sich, dass die vorgeschlagenen Maßnahmen nicht nur den Prozess der Planung unterstützen, sondern zudem die kontinuierliche Archivierung von Inhalten ermöglichen.

Abstract

The amount of digital content produced every day is increasing rapidly and challenges organisations with the obligation to preserve it for the long term. Preservation planning aims to select the best suited strategy to keep content accessible for a specific user community over time. This decision should be based on a repeatable process relying on objective criteria that leads to reproducible results and thus provides evidence to further trust. In the last decade a number of initiatives were dedicated to solve related problems on different levels, and provided insight in intricacies of digital preservation. Particularly, a preservation planning workflow was defined. It guides the planner to collect all related information, about the organisation, the content, requirements, and potential strategies. Experiments are conducted with representative samples and results are evaluated against specified requirements, providing the basis to take an informed decision and resulting in a well documented preservation plan. So far tool support is limited, and the lack of automation leaving it a mainly manual and effort intensive task. Information on content and the environment has to be gathered manually, and requirements are described ad-hoc or in a tool specific way, which hinders reuse and adoption for other purposes. Prototypes of registries for strategies and evaluation tools have been built, but they are hard to maintain and extend. And enacting the resulting preservation plan is only possible as proof of concept.

This thesis aims to improve the preservation planning process and increase automation by integration with its context. It identifies existing information sources and systems, adopts a common language based on a set of ontologies to describe related concepts, specifies missing interfaces to complete integration, and introduces a methodology to specify monitoring conditions. A prototype implementation of the planning tool is extended and evaluated in a case study. It shows that proposed measures not only benefit the planning process, but can enable continuous preservation of content.

Contents

List of Figures

List of Tables

List of Listings

CHAPTER 1

Introduction

The amount of digital content is steadily growing and poses new challenges to content holders, who have the need to preserve their digital assets for the long term. While bit preservation was tackled from the early beginnings of the computer age, only in the last decades attention was focused on the logical aspects of digital content and its environment. A number of endeavors world-wide and specifically in the European Union were dedicated to investigate related problems and potential organizational and technical solutions. This led to a deeper understanding of intricacies and challenges of digital preservation, and raised the awareness that the process of taking an informed decision requires a more structured approach.

Guidelines have been created to assist responsible organisations in defining their goals[Web03], usually captured as policies, and the preservation community tries to share this knowledge and build best practices[BSWW08]. These are usually high level statements formulated in natural language, but there are efforts to formalise them and make them accessible for automation[SM07]. Part of them relate to requirements on involved tools, and to preserve-worthy properties of the content, in literature often referred to as significant properties. Efforts are made to categorise and analyse these aspects of quality, as a standardised catalogue can foster reuse and enables automated evaluation[BR11], and provide the basis for analytical approaches[HB11][BKPR13].

Reference models provide the concepts to compare different archival information systems[ISO03] and ongoing efforts are made to improve interoperability between archives through standardised data exchange formats[1,2]. Recently also a set of open APIs has been proposed[Sch12] to enable integration of components with repositories.

A thorough knowledge about the content to be preserved is fundamental to decide on a preservation strategy, but the sheer amount of objects, their diversity and intrinsic complexity pose a challenge. Recently developed file identification and characterisation

[1] http://www.loc.gov/standards/mets/
[2] http://www.loc.gov/standards/premis/

tools provide more information and content profiling tools enable in-depth analysis of the collection and its characteristics[Pet13].

A variety of systems and tools were built, not all with preservation in mind, but nevertheless useful for this process. Advances in virtualisation technologies support emulation strategies, where not a new interpretation of the file is created, but the whole system necessary to render its information is preserved. Solution components range from tools to extract meta data or semantic information, to renderers to access the digital information, and migration tools, which transform objects to new representations. Workflows facilitate to compose such tools and provide a thorough documentation of used settings and the information flow, thus support repeatable execution with reproducible results. When applied to the content to keep it accessible for the long term we speak of preservation actions.

Selecting the most suitable preservation action for a specific scenario is challenging due to the number of potential candidates with their peculiarities, which have to be evaluated against the set of identified requirements and with sight to the diverse characteristics of the content, while considering constraints on resources.

1.1 Motivation and Problem Statement

Trustworthy decision making should be based on a repeatable process relying on objective criteria that leads to reproducible results and thus provides evidence. To this end a preservation planning workflow[SBNR07] was defined. It describes for each step the information to be collected: About the organization, the data itself together with representative sample objects, criteria which have to be met, and possible preservation solutions. Then the planner has to apply the preservation solutions to the sample objects and collect the outcomes. Finally for each preservation solution a score is calculated with respect to the defined criteria, and a recommendation can be provided. The preservation planning tool *Plato*[BKH08] was implemented as prototype to validate the workflow and make it easier accessible to planners. It guides the planner step by step through this workflow, and finally provides a well-documented preservation plan. A number of case studies proved workflow and tool useful. Nevertheless preservation planning is mainly a manual task and still an effort intensive endeavour:

1. Information describing the context where preservation takes place, such as institutional policies, has to be entered manually and is only retained as textual documentation.

2. Analysis of the collection and selection of representative samples is left to the planner, who then has to enter the findings manually in *Plato*.

3. Decision criteria are often created ad-hoc, hindering knowledge extraction and reuse. Moreover, this impedes automated evaluation and monitoring of operations.

4. As proof of concept registries for migration actions have been integrated, but they are limited and due to their closed nature they are hard to maintain and extend.

5. Prototype implementations for automatic evaluation of migration results with respect to the decision criteria are present, but only as part of *Plato* and therefore they are not available to ensure quality of results during operations.

6. Enacting the final preservation plan is only possible as proof of concept, for one repository only.

1.2 Aim of the work

As it can be seen in the problem statement, preservation planning is dependent on information from different sources and comprises a range of tasks. Due to its wide scope it is neither feasible nor advisable to cover all functionality directly. In the last years tools that can support this process were developed not only by the digital preservation community itself, but also for other purposes.

This work aims to resolve before mentioned issues, researching methods that allow to further automate the creation of preservation plans. It extends the existing planning component by integration of existing tools and systems through open interfaces, and facilitates knowledge sharing and reuse through adoption of a common language. Thus, it makes planning aware of the context in which it takes place to increase scalability of preservation planning.

My thesis statement is: Making preservation planning aware of its context by integrating adjacent systems and related information sources eases the preservation planning process, and creates new opportunities for automation. The aim of this work is to tackle before mentioned problems as summarized in the following goals:

G.1 Integrate organisational policies in a way that the embodied information is really used and actually drives the preservation planning process.

G.2 Facilitate describing the collection and selection of representative samples.

G.3 Foster reuse of decision criteria, improve automated quality assurance, and enable continuous monitoring of results according to the criteria determined during planning, by adopting a common language.

G.4 Integration with repositories through open interfaces, to improve automation and scalability.

1.3 Methodology

As a first step the preservation planning process itself is examined in detail to identify its drivers and information sources, and shortcomings. Then, current methods and approaches in scientific projects of relevant fields are researched.

The main part of this work is the adaptation of the planning tool's architecture: Ontologies are used to semantically integrate systems, and to annotate workflows which are used for experiments. Existing interfaces to preservation systems are integrated, and missing APIs to support management of preservation plans and monitoring of results are specified, including related data formats.

Finally, a case study to demonstrate the integration of the planning tool with the preservation environment is conducted to evaluate the achievements.

1.4 Impact

Parts of the work in this thesis have been published in conference articles:

- [BKPR13] described the results of analysing multiple decision cases, cross referencing criteria to gain insight on impact of specific criteria and criteria sets, that was enabled by my work on adapting *Plato* to the quality ontology, and translating the existing knowledge base.
- [KPD$^+$13] presented a first integration of *Plato* with content profiling and adopting ontologies for policies and quality, supporting a preservation environment.
- [DKK$^+$14] demonstrated the full integration with a preservation environment, adding integration with repositories based on open APIs, and monitoring of results.

1.5 Structure of the Work

The remaining part of this work is structured as follows: Chapter 2 gives an introduction to digital preservation, particularly preservation planning, provides the background on fields which this work builds upon, and concludes with a summary highlighting the gaps. Chapter 3 presents a set of ontologies that can be used to describe policies and properties of quality, and shows how they can drive and support preservation planning. Integration with content profiling, repositories, monitoring systems and operations is discussed in Chapter 4, and Chapter 5 introduces a methodology to derive monitoring conditions, before presented changes are evaluated based on a case study in Chapter 6. Finally Chapter 7 summarises contributions, and results and future works are discussed.

CHAPTER 2

Background

2.1 Digital Preservation

The amount of digital content produced every day is increasing, and nowadays large part of its objects are born digital, without physical representations. This poses new challenges to content holders, who have the need to preserve their digital assets for the long term.

Bit preservation was tackled from the early beginnings of the computer age. One of its problems that became apparent almost immediately is decay of storage media over time. Its pace of degradation depends on the type of storage media and environmental conditions and might lead to total information loss. New technologies aim to improve data density, access times, and durability of media, but ironically advances in storage technology pose a threat too: currently common storage media are likely to be outdated in a couple of years and replaced with newer ones, with the risk that media together with appropriate reading devices are going scarce over time. Therefore the data has to be periodically copied to a new medium of the same type, which is referred to as *refreshing*, or *migrated* to a medium of a newer generation. Replicas, multiple copies preferably distributed to different locations, reduce the risk to loose access in case of data corruption, but it is necessary to audit these copies - periodically check them for errors - and keep track of the different copies. Hence, while counter measures are known, the problem itself cannot be seen as solved, as they come with high costs in terms of processing power and money, and advance of technical solutions cannot keep pace with the increasing data volume to be preserved[Ros10].

Even more, preserving the bitstream is not sufficient and information on how to interpret and access it is necessary, which Rothenberg illustrates very well[Rot95]. The most prominent preservation strategies are *technology emulation* and *information migration.*

The former aims to preserve not only the objects, but also its original application program by mimicking the underlying hardware and software dependencies, and thus pertaining the original look and feel of them. Rothenberg suggest a system based on an emulation specification interpreter which runs on a virtual machine, such that only the virtual machine has to be ported to future hardware platforms, and digital objects are preserved together with the software and its corresponding emulation specification to be instantiated when needed[Rot99]. However, this idea was criticized as being expensive, and postponing the creation of an emulator would make it impossible to test its performance, as the resulting rendering cannot be compared to the original any more. In general the dependencies of applications can be complex, to provide properly configured environments is related with costs, and access to objects through emulation is slow. Nevertheless, emulation is especially suited for preserving applications or interactive objects like video games[GBR08].

Information migration, on the other hand, aims to transform the intellectual content to another representation which is more stable or can be accessed with current technology. Preferred are sustainable target formats that are well documented, not restricted by intellectual property rights, and standardised. This conversion is inherently related to information degradation, because most likely different formats do not cover the same functionality. Even for migrations between different versions of the same format reversing the process might be difficult, because features might have been discontinued, or the process involved lossy conversions in the first place. Therefore, an assessment of the significant properties of the content is necessary, the expected level of quality has to be defined, and quality assurance mechanisms have to be in place to detect unforeseen losses.

An in depth discussion of preservation strategies and initial digital preservation endeavours is given in [LSL$^+$02]. Current attempts focus on preservation of more complex objects,like business processes[1] and workflow based scientific experiments in data-intensive science[2], or web archiving.

A number of endeavours world-wide and specifically in the European Union were dedicated to investigate related problems and potential organizational and technical solutions. The OAIS reference model[ISO03] provides a high level description of the functional entities of a preservation systems and their responsibilities, how they interact, and the information exchanged. To evaluate the ability of institutions to preserve entrusted content and establish a measure of trust the Trustworthy Repository Audit & Certification: Criteria and Checklist (TRAC)[3] provides criteria catalogues and check-lists for self assessment for repositories, and the Digital Repository Audit Method Based on Risk Assessment (DRAMBORA)[4] adapts standard risk-management models to the domain of repositories. Others used these tools to analyse existing information systems, and focus on their capabilities[5].

[1]http://timbusproject.net/
[2]http://www.wf4ever-project.org/
[3]https://www.crl.edu/sites/default/files/d6/attachments/pages/trac_0.pdf
[4]http://www.repositoryaudit.eu/
[5]http://www.shaman-ip.eu/

2.2 Preservation Planning

One of the core functional entities of the OAIS reference model is *Preservation Planning*. At its center is a component to *Develop Preservation Strategies and Standards*, which is informed by components to *Monitor Technology* and *Monitor Designated Community.* The resulting proposals and recommendations support the administration of the information system to deal with identified risks and opportunities.

These functions are described only on a high level, it is not mentioned how they can be achieved. They could be provided by human personnel relying on standard software like text processors and spreadsheet software, but the results of this manual approach inherently lack proper documentation of the process and such of trustworthiness. Their quality depends highly on the expertise of the responsible planner.

The main goal of preservation planning is to select the best strategy to preserve the content in question for a specific group of users (Designated Community). In order to fulfil requirements of maintaining a trustworthy repository and providing the evidence needed for risk management tools like TRAC and DRAMBORA, this process should be repeatable, the results reproducible, and the procedure and outcomes well documented. Potential solutions cannot be compared directly, and to enable repeatable and reproducible results objective and measurable criteria have to be defined. These should cover properties of the content that should be preserved, requirements and performance of tools, and characteristics of formats; resulting in a heterogeneous set of potentially conflicting requirements - for example keeping storage costs low, but retaining uncompressed objects with high resolution - which have to be reconciled to determine the best suitable solution. At the same time the subjective preferences of the organisation and the designated community have to be taken into account.

Therefore it can be seen as instance of the multi criteria decision making problem to solve component selection. Two aspects have to be considered separately in the light of a specific scenario: the value of potential measurements of one criterion alone, and that of different criteria compared to each other.

Typically measurements of these criteria differ in scale and extent and thus are incommensurable. One way to integrate them is the use of **utility functions** to transform measurements to target values of a uniform scale of 0 to *max_value*, where 0 represents an unacceptable value, and *max_value* the highest reachable score.

Subjective preferences for certain criteria over others have to be reflected independently of this, and can be modelled defining **weights** for each criteria. Finally overall scores can be calculated by aggregating target values considering their weights.

2.2.1 Preservation Planning Workflow

Trustworthy decision making should be based on objective criteria and a repeatable process that leads to reproducible results and thus provides evidence. The project Preservation and Long-Term Access Through Networked Services (PLANETS) aimed to provide long term access to digital assets by defining an architecture based on networked services, and build tools to support preservation processes. Part of this endeavour was to identify the information needed that drive preservation processes and specify how to collect it. [HBS+08], as seen in [BKG+09], defines a preservation plan as follows:

> A preservation plan defines a series of preservation actions to be taken by a responsible institution due to an identified risk for a given set of digital objects or records (called collection). The Preservation Plan takes into account the preservation policies, legal obligations, organisational and technical constraints, user requirements and preservation goals and describes the preservation context, the evaluated preservation strategies and the resulting decision for one strategy, including the reasoning for the decision. It also specifies a series of steps or actions (called preservation action plan) along with responsibilities and rules and conditions for execution on the collection. Provided that the actions and their deployment as well as the technical environment allow it, this action plan is an executable workflow definition.

A preservation planning workflow[SBNR07] was defined to formalize the process of establishing a preservation strategy. Figure 2.1 gives an overview of the different steps of this workflow. It describes step by step the information to be collected, in four main phases: In **Define requirements** information about the organization, the data itself together with representative sample objects, and criteria which have to be met have to be specified. In **Evaluate alternatives** possible preservation actions have to be determined, then the planner has to apply these actions to the data and collect the outcomes. In **Analyse results** the outcomes are unified and post-processed, such that for each action a score can be calculated with respect to the defined criteria, and a recommendation can be given. Finally, after deciding for one preservation action, in **Build preservation plan** a preservation plan is created, which, once validated, can be implemented in the repository.

Define Requirements

In the first phase all requirements and restrictions shaping the decision have to be collected. Starting with step **Define basis**, the context of the preservation process has to be described. For further reference and management of plans an identifier is required; a description of the plan, the responsible planners and the organisation complete information to enable accountability, a key element of trustworthiness. Furthermore organisational objectives and constraints, e.g. the mission statement or financial constraints, have to be recorded. These drivers can be expressed as organisational policies, which will be discussed in more detail in Section 2.4. Depending on the designated community the objects are preserved for, different requirements on access might arise. Imagine the case of preserving photos, preserving the image alone might be enough for some, but meta-data stored along the image data could be highly valuable for others. For more complex objects like video games the different preferences can be more diverse: while for some a series of screen-shots might be sufficient, others might attach importance to animations and sound, requiring more ample derivatives. Hence, describing for whom the

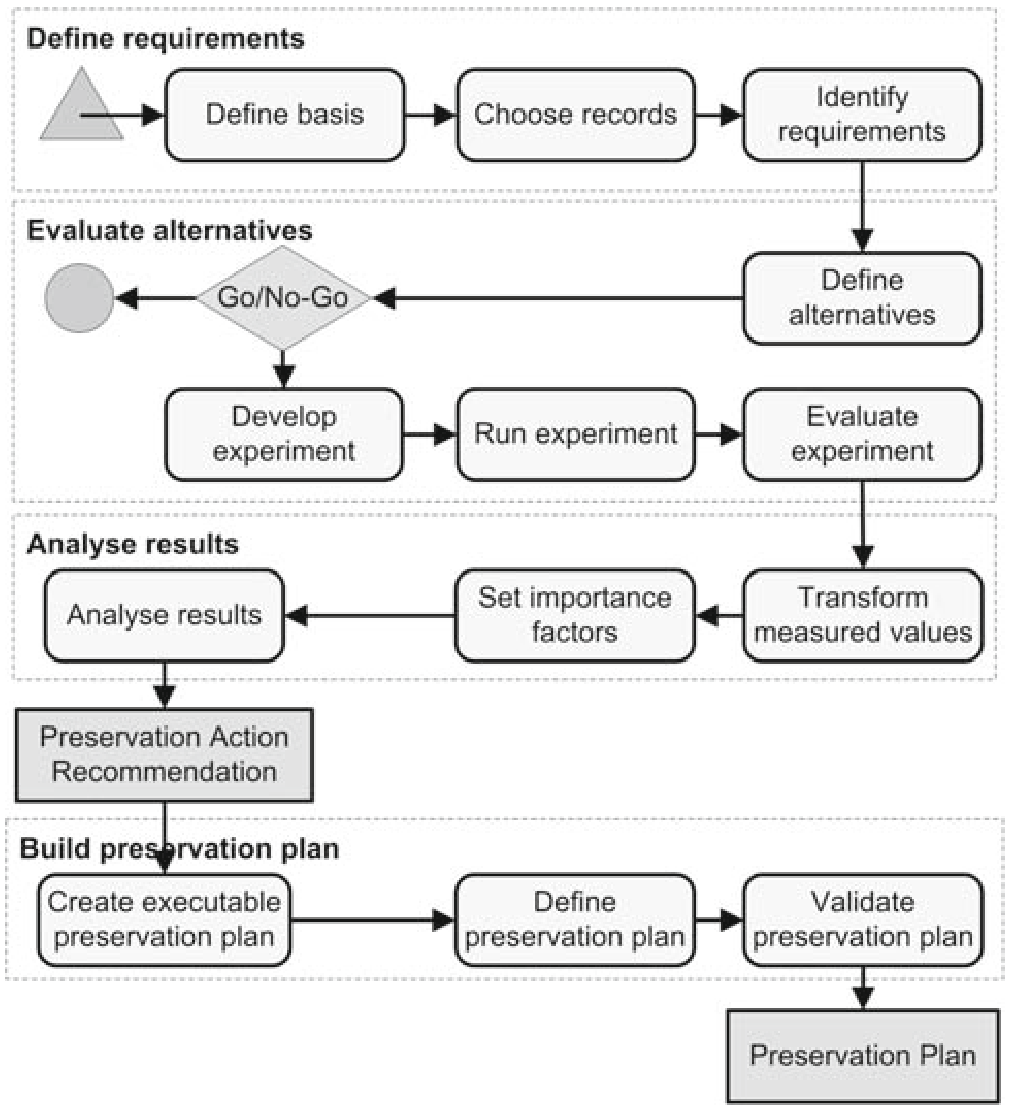


Figure 2.1: Extended Preservation planning workflow [BKG$^+$09]

objects should be preserved for can help to determine important requirements in later steps.

In step **Choose records** the collection of objects to be preserved has to be described. This includes characteristics like the type of objects, if it is a heterogeneous collection or a homogeneous set, the number of objects and the expected growth rate, but also the time period for which the objects should be preserved. Determining these features requires a detailed analysis of the collection referred to as *content profiling*. As the number of objects is usually too high to run experiments on all of them, a representative set of sample objects has to be selected. These samples are then analysed in-depth via characterisation tools and the extracted features are stored in the plan. To keep the effort for both - later experiments and describing the objects - low, the set of samples should be as small as possible, while still covering the important features. Due to the high number of objects on the one hand, and the inherent complexity of file formats on the other hand, content profiling and sample selection are complex and effort intensive tasks. Yet this step is crucial, as superficially selected samples could prevent the detection of problems with certain preservation actions and thus rendering part of the experiments useless, potentially leading to the selection of a faulty preservation action and finally data loss. Owing to their importance both are discussed in more detail in Section 2.3.

In step **Identify requirements** all requirements on the preserved objects, but also on the process and used tools have to be collected. These are based on the information gathered so far - the mission statement, financial constraints, organisational policies - and shaped by the designated community and the objects to be preserved. Usually the former are high level statements formulated in natural language, which makes it difficult to assess the degree of conformance or to compare different preservation actions objectively. Therefore the goal here is to break these high level objectives down to measurable criteria. This is not a straight forward task, as it is not always easy to specify the essence of an object, what of its characteristics are of importance to be preserved, or describe its qualities in a quantifiable way. Naturally, it is desirable to reuse and share knowledge gained from previously done similar scenarios, but this requires a controlled way to specify properties of quality. Section 2.6 discusses endeavours so far. Following this approach results in the so called *objective tree*, which provides the basis for the evaluation in the subsequent phases.

Evaluate Alternatives

Once the requirements on the preservation process are clearly defined, available preservation strategies have to be identified, described and evaluated.

First step in this phase, **Define alternatives**, is to identify potential preservation strategies for the content described in step Choose records. This requires an in depth search for tools able to handle the content. Software registries can support this task as well as knowledge exchange with the preservation community. Identified solutions should be described in detail and documented in the plan to strengthen evidence. This includes their requirements on the platform, operating system, required libraries, and existing manuals. Note that also maintaining the status quo could be a viable alternative, if no appropriate solution is available, or currently the funding is insufficient.

Step **Go/no-go** is an intermediate step before starting with the evaluation. Here all gathered information can be reviewed. Financial limitations or lack of access to the software could prevent the planning process to be continued. In this case single alternatives can be discarded from the process, and the reasoning recorded.

In step **Develop experiment** for each alternative the procedure of applying the alternative to the content has to be described in detail. Beside the technical environment and used parameter configurations of tools also the procedures of gathering the results or measuring runtime statistics have to be documented to enable reproducibility of results.

When the experiment set-up is complete, in **Run experiments** the preservation solutions described in the previous step are applied to each of the sample objects. For migration actions this results in an outcome which can be stored in the plan along with eventual log files. Emulation actions require to take comprehensive notes of the emulation run.

To conclude this phase, in **Evaluate experiment** the alternatives are evaluated against the criteria defined in step *Identify requirements*. For criteria related to the alternative itself, like costs of adopting this strategy, or information about the involved tools like applying licenses, it is sufficient to provide these values once per action. However, criteria related to the outcome of an alternative, that is to the file format, characteristics of the file itself, or the process, have to be evaluated per alternative and sample, causing fast growing effort with increasing number of the both. These values are again stored in the plan and provide evidence for the decisions taken in the following phases.

Analyse Results

After the requirements have been defined and the experiments conducted, the values collected in the previous step are analysed to determine the best solution.

As mentioned before these measurements can be of different scales: Considered are numeric values of different value ranges, ordinal values, and nominal values without imposed natural order, therefore a ranking of alternatives cannot be calculated directly. In step **Transform measured values** for each criteria a utility function has to be provided that maps the values to a target value between 0 and 5.

Naturally, not all criteria are of equal importance for a scenario. **Set importance factors** allows to distribute the weight between the siblings of each node of the objective tree, because this should separated from defining the utility function to make such a decision clearly visible.

After this, **Analyse results** calculates the ranking by multiplying each target value by its weight and aggregating the products for each alternative. First this is done by multiplication of all products from the leaves of the objective tree up to the root. An unacceptable target value will render all products along this path up to the root to 0, which makes it easy for the planner to get a quick overview about unsuitable alternatives and the origin of their problems. Then the sum of all products is calculated, which allows to rank the alternatives according to this score. Sensitivity analysis is applied to make cases visible, where slight variations of the weighting would lead to different results, and a final ranking together with the highest ranked alternative as *Preservation Action Recommendation* is presented to the planner. Note that the planner has the responsibility to take the final decision for an alternative and still has the opportunity to select an other preservation action than the suggested one. Anyway, then the reasoning behind this decision and potential effects of adopting the strategy have to be documented in the plan.

Build Preservation Plan

After a preservation action has been chosen, a preservation plan can be built. Step **Create executable plan** defines everything needed to carry out the preservation plan. These include the triggers and conditions under which the plan is executed, the environment of the preservation action, parameter settings of involved tools, and the subset of the collection the plan is applied to. Moreover a subset of the criteria used for evaluation can be selected for quality assurance to make sure the defined limits of critical aspects are also met during application of the preservation action. This information is stored in the so called *preservation action plan*.

In step **Define preservation plan** non technical details of the preservation plan are defined. This includes a more accurate estimation of the costs adopting either the lifecost or Total Cost of Ownership model and details to monitor adoption of the preservation plan: Responsible persons for execution and monitoring have to be defined, and the triggers which require to revise the preservation plan, like a mandatory periodic review or changes in the environment.

Finally, in step **Validate plan** all the evidence collected, the taken decision, and the impact on the organisation have to be reviewed. Typically this is done by a different role than the planner. Once a plan has been approved, it should not be changed without starting a formal revision process.

This process results in a fully documented plan: starting from the requirements and obligations of the organisation, it contains all considered alternatives, detailed descriptions of the experiments allow to repeat and verify the results, and concrete measurements which led to the final decision are recorded as evidence.

2.2.2 Preservation Planning Systems

The prominent position of the preservation planning module in the OAIS reference model led to integration of such modules also in commercial preservation systems. These tools are by nature closed, with vendor specific solutions and knowledge bases, and implement rather ad hoc workflows. So far they lack automation of experiments and QA.

The preservation planning function of the *Active Preservation Framework* developed by The National Archives (United Kingdom) is based on the technical registry *PRONOM*[6], with the focus on file formats and related risks[Bro08]. *Rosetta*[7], a preservation system developed by ExLibris, provides a module for preservation planning which follows a simple worflow of risk analysis, evaluating migration paths, and applying the preservation action.

The preservation planning tool *Plato*[BKH08] was implemented as prototype to validate the workflow discussed before and makes it accessible to planners. The web application guides the planner step by step through this workflow, and finally provides a well-documented preservation plan. The context of the preservation planning process, the organisation and its goals and the designated community have to be described manually.

[6]`https://www.nationalarchives.gov.uk/PRONOM/default.htm`

[7]`http://www.exlibrisgroup.com/category/RosettaOverview`

This textual documentation is used as evidence and can be consulted later, but does not facilitate the further planning process.

The collection has to be analysed externally to get an overview on its content and to determine representative sample objects. This information has to be introduced to *Plato* manually. Even more the samples have to be added and described one by one. In case they are uploaded to enable later automated experiments and reproducible results for improved evidence, *Plato* runs internal characterisation tools on the files and stores the extracted properties along with the samples.

In step *Identify Requirements* a graphical representation of the objective tree is provided, which allows to specify criteria and structure them adding inner nodes. It also provides an import of mind maps defined with *Freemind*[8] to facilitate collaboration. It incorporates the quality model discussed in Section 2.6 but due to its closed nature and incomplete state decision criteria are still often created ad-hoc. This hinders knowledge extraction and reuse, because different preferences in naming might lead to redefinitions of actually already existing measures, and manual data cleaning is necessary. Moreover it impedes automated evaluation and as a result continuous quality assurance.

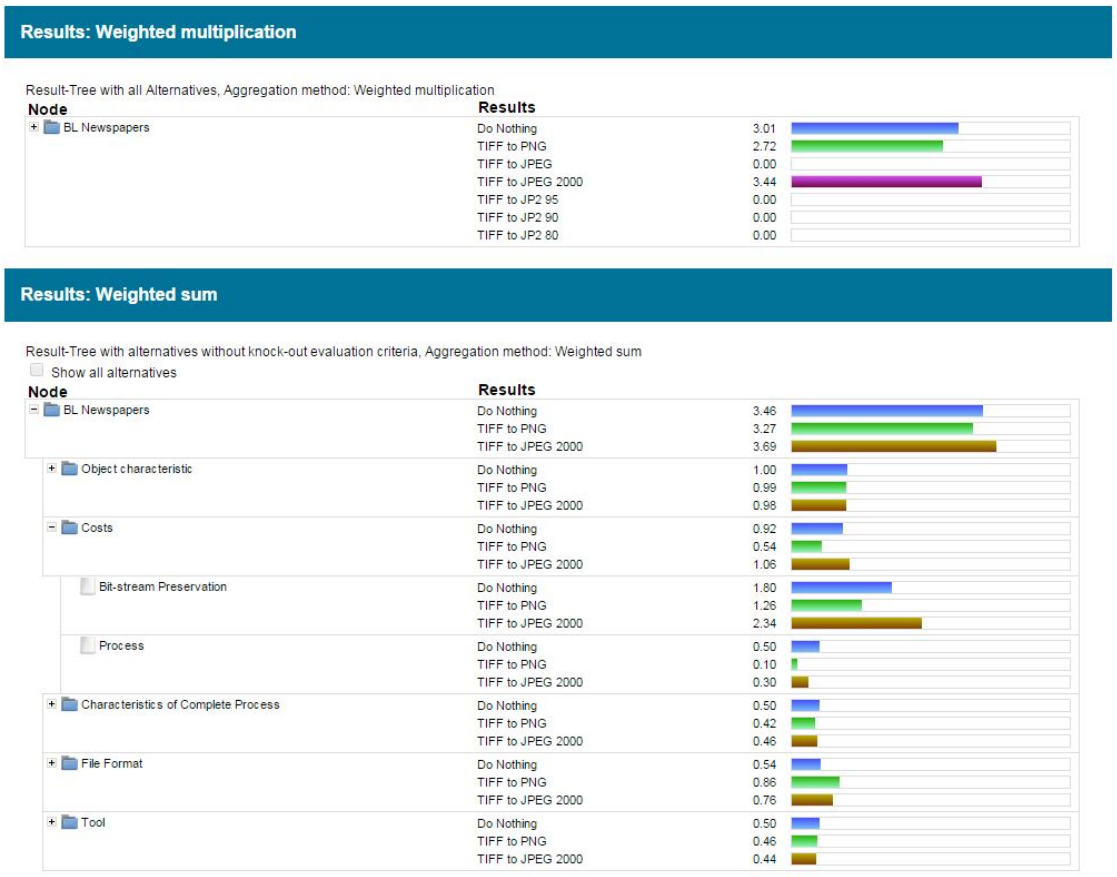
To define alternatives, beside manually describing an alternative, *Plato* integrates a range of registries which can be queried for possible solutions. The tool registry *miniMEE* provides a list of migration and emulation services that can be invoked, and returns also information on their runtime behaviour [BKK$^+$09]. It is a proof of concept, does not consider scalability, and is difficult to configure. The *P2* semantic registry can be queried for information on solutions [THC11]. The Planets service registry can be used to look up web services deployed in the Planets interoperability platform. Its configuration and maintenance requires substantial effort and technical understanding. All these registries are specifically built for preservation purposes, but hard to keep up to date and to extend.

Experiments for alternatives based on Planets web-services and tools from *miniMEE* can be run in an automated way. For evaluation of the experiment results *Plato* uses a prioritised list of evaluators which are run sequentially, where each evaluator can complement still missing measurements. *XCL* and *FITS* are used to extract characteristics from the files, and *P2* is used for information on software and formats in general, other evaluators are added as proof of concept, for example an evaluator for image comparison based on *ImageMagick*[9]. The focus is on evaluation during planning and this integration of tools for evaluating the results is hard wired, therefore they are not available for continuous quality assurance.

Figure 2.2 shows how the aggregated evaluation results are visually represented to provide a quick overview of the performance of alternatives. From seven alternatives under test four have unacceptable values and are therefore excluded, for the remaining three alternatives the scores are aggregated and can be examined in detail.

There exists a proof of concept implementation to generate an executable plan with a selected migration service, but it can be run only on one specific repository.

[8]`http://freemind.sourceforge.net/`
[9]`http://www.imagemagick.org/`



Results: Weighted multiplication

Result-Tree with all Alternatives, Aggregation method: Weighted multiplication

Node	Results	
BL Newspapers	Do Nothing	3.01
	TIFF to PNG	2.72
	TIFF to JPEG	0.00
	TIFF to JPEG 2000	3.44
	TIFF to JP2 95	0.00
	TIFF to JP2 90	0.00
	TIFF to JP2 80	0.00

Results: Weighted sum

Result-Tree with alternatives without knock-out evaluation criteria, Aggregation method: Weighted sum

Show all alternatives

Node	Results	
BL Newspapers	Do Nothing	3.46
	TIFF to PNG	3.27
	TIFF to JPEG 2000	3.69
Object characteristic	Do Nothing	1.00
	TIFF to PNG	0.99
	TIFF to JPEG 2000	0.98
Costs	Do Nothing	0.92
	TIFF to PNG	0.54
	TIFF to JPEG 2000	1.06
Bit-stream Preservation	Do Nothing	1.80
	TIFF to PNG	1.26
	TIFF to JPEG 2000	2.34
Process	Do Nothing	0.50
	TIFF to PNG	0.10
	TIFF to JPEG 2000	0.30
Characteristics of Complete Process	Do Nothing	0.50
	TIFF to PNG	0.42
	TIFF to JPEG 2000	0.46
File Format	Do Nothing	0.54
	TIFF to PNG	0.86
	TIFF to JPEG 2000	0.76
Tool	Do Nothing	0.50
	TIFF to PNG	0.46
	TIFF to JPEG 2000	0.44

Figure 2.2: Screenshot of *Plato 3* - Analyse Results

All in all it can be seen that preservation planning is still an effort intensive endeavour with mainly manual processes. Despite of this attention of the preservation community is growing, and a number of real world case studies proved workflow and tool useful. Applications range from preserving digitised documents[KRK$^+$09], preservation of video games[GBR08], to database archiving.

2.3 Content profiling

Thorough knowledge about the content of the collection to be preserved is fundamental to decide on a preservation strategy. To gain insight and understand its complexity and peculiarities meta data on its objects is required.

File identification determines the format of a file (e.g. PDF or Jpeg) and its version, whereas file characterisation extracts more specific information like file size and creation time, but also intrinsic features like number of pages of documents or used compression algorithms.

In the last years a range of existing tools have been identified as useful by the preservation community, e.g. the *Exiftool*[10]. And new tools have been developed from scratch: the *Digital Record Object Identification* tool (*DROID*)[11] supports file identification and provides Pronom unique identifiers, *Apache Tika*[12] is conducting extended meta data extraction.

Others like the *File Information Tool Set* (*FITS*)[13] take a different approach and build on existing tools, invoking them following predefined chains, and unifying their outputs.

The analysis of a collection of several hundreds of thousands of objects is time and resource intensive, therefore it is tempting to restrict it to identification, as proposed by Hitchcock and Tarrent[HT11]. This might be viable as a first step for heterogeneous collections, but Petrov shows that in general this is not sufficient, as even files of the same file format can vary greatly, and simple features like file size can have a great impact on preservation tools[Pet13].

He presents a software prototype, Clever Crafty Content Profiling of Objects (*c3po*)[14], which aggregates characterisation data, allows filtering for specific characteristics in a scalable way and to calculate representative sets of sample objects. A command line utility can process characterisation output of *Apache Tika* and *FITS*, and distils it in a form suitable for monitoring systems like *Scout*, which will be discussed in Section 2.5.2. A web front-end visualises the data and allows to explore it interactively. The distribution of digital object characteristics can be analysed and used to create sub-sets with certain properties, using interactive filtering on the charts themselves. Different algorithms are available to calculate representative samples. This aggregated information can then be exported as a content profile and used for preservation planning.

2.4 Organizational objectives and constraints

The reasons organisations have to preserve their digital assets are manifold. In industry legal obligations might require to keep plans and documentation of design processes accessible, for scientific institutions results of experiments might be the foundation of their leadership in a field and credibility when it comes to reproduce findings, or the organisation already had the mandate to preserve non digital heritage and is therefore thought to protect digital heritage too.

Guidelines like furthered by UNESCO[Web03] help organisations to define their role and actively embrace their mandate to preserve digital heritage, and concretise their goals and regulations. An active community tries to share this knowledge and build a body of best practices on how to define preservation policies.

[10]`http://www.sno.phy.queensu.ca/~phil/exiftool/`

[11]`http://www.nationalarchives.gov.uk/information-management/manage-information/preserving-digital-records/droid/`

[12]`http://tika.apache.org/`

[13]`https://code.google.com/p/fits/`

[14]`http://peshkira.github.io/c3po`

These provide the context of the preservation process, and guide decisions taken to achieve these goals. Yet, they usually consist of high level statements describing strategies and commitments of the organisation, as on the basis of the ISO 16363 Repository and Audit and Certification catalogue[ISO10]. A study on preservation policies analysed existing policies of organisations and aims to create guidelines for writing own policies[BSWW08], but they are as well complex to implement, not least due to the ambiguity of natural language. As such, they correspond more to what the Object Management Group refers to as *business policies*: *elements of governance* that are *not directly enforceable* and *exist to govern*[Obj08], - means to express drivers and constraints and resulting goals and objectives.

Smith and Moores argue that while archivists work primarily at the policy level, preservation systems work at the rules level, and therefore preservation policies first have to be translated to rules that can then be enforced by rules engines[SM07]. In this sense high level policies are first translated to so called concrete policies, which in turn can be translated to rules and mapped to capabilities of the system. They further suggest that a standard language to express concrete policies could help to share these statements between different preservation environments or parts thereof, and serve as an intermediate format. Finally, the capability of systems to enforce these rules can be automatically evaluated, and the coverage provides an indicator about the trustworthiness of the preservation system. They focus on general operations of an archive, and the given example shows a policy as manifestation of already taken decisions.

So far there exists no standard to formulate preservation policies that considers preservation as a continuous process that requires adaptation, which includes planning.

2.5 Adjacent systems

Beside the organisational setting preservation planning depends on information about the content, usually managed in repositories. It produces a plan which should be implemented in the repository, together with conditions which have to be monitored.

2.5.1 Repository

The OAIS model developed by the Consultative Committee for Space Data Systems and later adopted by ISO as reference model for Open Archival Information Systems [ISO03] gives a high level description of a preservation system. It describes the information exchanged, the necessary functional entities and the actors and how they interact.

More importantly, it provides a terminology for these concepts and thus a basis to discuss and compare different information systems. Yet, the reference model does not impose any implementation details, but describes only the functionality which is necessary to preserve digital content over an undefined period of time. Hence, while technical solutions can support the described functions, they are not required to adhere to this standard.

There are a number of open source products, like *RODA*[15], *DSpace*[16], *fedora commons*[17], and *EPrints*[18], and commercial products like *Rosetta*[19]. Even though the majority of repositories adhere to the OAIS standard, this is more due to its generality than to their actual architectural similarity or compatibility.

But there are attempts for standardisation: The METS (Meta Data Encoding and Transmission Standard) [20] was developed to ease exchange of meta data between repositories and ingest of digital objects to repositories. While it is supported by more and more repositories, standardised interfaces for ingest and retrieval do not exist. This hinders the development and reuse of services among repositories.

The SCAPE Preservation Platform is designed to provide a scalable hardware and software infrastructure for execution of preservation actions[Sch12]. It consists of two main entities: The Execution Platform with support for deployment of tools and running them, and the Digital Object Repository that provides interfaces for data management, that is: retrieving and storing digital objects and manipulating their meta data. The focus is on the definition of open APIs to allow integration with other preservation systems. Until now they are adopted by *RODA*, and partly implemented in *Fedora 4* and *EPrints*.

2.5.2 Monitoring Systems

Digital preservation aims to keep digital assets accessible over a period of time, despite physical threats, advances in technologies and changing trends. Preservation planning compares currently available solutions and requirements to determine the best suited strategy. It is necessary to continuously monitor changes of the environment, check if expectations are still met, and adjust procedures if required.

Part of the functional entity Preservation Planning of the OAIS model are the two components *Monitor Designated Community* and *Monitor Technology*. The first tracks changes in service requirements and available technologies of producers and consumers. The latter traces “emerging digital technologies, information standards and computing platforms (i.e., hardware and software) to identify technologies which could cause obsolescence”[ISO03].

Becker et al. stress that this task is usually carried out manually and on a higher level, when it should be related directly to the objectives discovered during planning [BDF+12]. They analyse the requirements on a monitoring system, identify potential information sources and propose a high level architecture of a preservation watch system that combines information from different sources and enables definition of questions which can trigger events.

[15]http://www.roda-community.org/
[16]http://www.dspace.org
[17]http://fedora-commons.org/
[18]http://www.eprints.org/software/
[19]http://www.exlibrisgroup.com/category/RosettaOverview
[20]http://www.loc.gov/standards/mets/mets-tools.html

Scout[21] is a prototype implementation of such a preservation watch system, and monitors the preservation system and its environment [FPD+12]. Information from different sources is collected via so called source adaptors. They process the information and translate it to the internal knowledge base of *Scout*.

This is modelled as simple ontology, instantiating a model of entity-, property-, value relationships. Values are extended with information on time of measurement to model trends, and its source to maintain provenance of values. It supports reasoning, and watch conditions can be defined as queries that are executed periodically and can trigger email notifications.

2.5.3 Systems and Workflows

During preservation planning potential preservation strategies are evaluated. This includes the application of emulation or migration tools to sample objects. In case of migration quality assurance has to be applied to the resulting files, they have to be identified and characterised to extract relevant properties. In the end the selected solution will be applied to the whole content.

This requires knowledge about available tools, and installation and setup of these tools require technical background and effort. The *P2* semantical registry uses linked data principles to combine information from different sources like *PRONOM* and *dbPedia*[22] and can be queried for solutions [THC11], but only to inform. One of the first attempts to provide a registry with predefined migration tools is *CRiB*, which wraps command line tools as web services that can be looked up and invoked[FBR07]. It also uses the common interface of its services to generate new migration paths by chaining multiple services together. The SCAPE action services [23] provide a range of tools installable as Debian packages.

A key requirement for trustworthy decision making is reproducibility of results. Hence the experiment setup, versions of involved tools, their dependencies and parameter settings have to be described in detail.

A Workflow describes a sequence of operations to process information or resources in a reproducible way. Graphical user interfaces support creation and composition, thus making them accessible to technically less versed.

Taverna[24] is a workflow management system for scientific workflows. It allows the invocation of web services as well as command line tools, scripts, and functionality from Java libraries. It provides a GUI to create and compose workflows, and the possibility to run workflows either locally or on a dedicated server[HWS+06]. Even though first adopted for bioinformatics and astronomical experiments, it is domain independent. MyExperiment[25] is a social workflow sharing platform for predominant *Taverna* workflows. Networks of interest can be built, and workflows can be published for

[21]`http://github.com/openpreserve/scout`
[22]`http://wiki.dbpedia.org/`
[23]`http://github.com/openplanets/scape/tree/master/pc-as`
[24]`http://www.taverna.org.uk`
[25]`http://www.myexperiment.org/`

sets of members, together with sample data, descriptions, and meta-data to ease lookup. Hence it provides the infrastructure for controlled sharing that maintains authorship and provenance[RGS09].

2.6 Quality and Measures

In Section 2.2 preservation planning and its central goal of selecting the best suited preservation action is discussed. The presented approach uses multi criteria decision making based on utility analysis, which requires to define measurable criteria for the evaluation of potential solutions. Identifying relevant criteria is a crucial step and decides over the quality of the final decision. Relevant characteristics of the objects that should be preserved have to be identified and described, which is challenging and requires experience. The requirements on tools and their performance, and characteristics of formats have to be considered as well. For this reason it is suggested that workshops with stakeholders can be useful to find relevant criteria, and at the beginning preferably a preservation expert should be present [BKG+09].

First case studies started to collect and define criteria manually, and early versions of *Plato* provided tool support to organise and structure the tree with objectives, and provided a set of scales to choose from. Even though the lack of guidance sometimes led to ill-defined criteria due to problems of finding suitable measurable criteria for properties of interests, the analysis of scenarios for different content types and settings resulted in a valuable body of knowledge. Specifying criteria from individual case to individual case turned out to be problematic.

First it hinders knowledge sharing. Once gained insight into a certain scenario can only be reused by passing on the whole plan. Comparing different scenarios is effort intensive, as the criteria have to be aligned manually.

Second, identifying the requirements is an intellectual task and cannot be automated itself, but it directly affects evaluation of alternatives. Collecting the measurements for the criteria constitutes a big part of the planning effort and holds potential for automation, which cannot be seized with manually defined criteria. Moreover, once a preservation action was adopted, its performance has to be monitored to make sure that the expected level of quality is not only met for the selected set of representative samples, but for the whole content. This requires a mean to specify statements on quality in a machine enact-able way.

To address the before mentioned problems the authors of [BR11] propose a taxonomy of criteria for the digital preservation domain with sight to automated evaluation. It is based on a set of completed case studies conducted with *Plato*, and results in a criteria catalogue consisting of 500 criteria suitable for different scenarios. This taxonomy is organised according to the sources of measurements, and can be used to inform an evaluation framework. Examples are given how different registries can be used to provide static information, and characterisation services or systems for controlled experimentation to retrieve dynamic measurements.

The ISO 25010 standard: Systems and software engineering - Systems and software Quality Requirements and Evaluation (SQuaRE) describes a hierarchy of high level quality attributes, integrating characteristics about *software in use*, the outcome of interaction, with characteristics about static properties of the software, and dynamic properties of the system in use. This quality model can be used to reorganise the taxonomy described before, and can provide a point of reference in case of doubt about meaning of certain quality attributes [HB11]. The resulting catalogue is used to analyse the importance of criteria and their impact on the final decision.

2.7 Gap Analysis

The previous sections described the context of preservation planning and the systems and tools involved in the preservation process.

What is missing to utilise information sources and systems?

First, we can observe that there is a common information need:

- Organisations need to express their goals and policies
- Solution components for identification, characterisation and migration of objects operate on specific types of files, and output information on these or new representations. They are often not created or maintained by the preservation community itself, but need to be described.
- Content profiling tools use information on file types and characteristics to analyse them and provide aggregated statistics to gain further insight on the content.
- *Plato* needs to specify the requirements on preservation components, look up suitable candidates, and components for evaluation of the results, and finally describe the executable plan and quality assurance metrics.
- Monitoring systems need to watch properties of environments and operations, and require means to specify rules about them and trigger events in case they apply.

These components all work with or produce the same kind of information, but lack of a standardized input and output data that in most cases is only semi structured. This hinders automated processing of the results or even composition of components.

Second, recently new systems for content profiling and monitoring and open interfaces to interact with repositories have been developed, and require architectural changes to be integrated.

The following chapters discuss necessary changes and extensions to the planning component, before in Chapter 6 the results are evaluated.

CHAPTER 3

Data- and Software Quality Model

This chapter discusses how preservation policies and requirements definition can be supported within Plato.

First the related problems are outlined. Then an ontology to formulate policies is presented. Its integration in Plato is described, and how it directly benefits the definition of requirements on preservation processes. In later chapters we will see how it can be used to describe components, improve automated evaluation, and to specify statements of quality for monitoring.

3.1 Problem statement

Section 2.7 pointed out a lack of collaboration between tools, and that there is a common information need in the systems discussed so far.

Organisations have to express their goals and policies. These describe the organisational setting, the content in custody, the designated community it is preserved for, and can contain statements about how the content is accessed, requirements on involved tools, formats and their characteristics that should be endorsed or avoided, and important aspects about the content that have to be preserved.

During planning the organisational setting has to be documented, and the content described. Preservation actions have to be discovered that are suitable for the content in question and conform to requirements, e.g. on the runtime environment or applying licensing schemes. As basis for an objective comparison requirements on the process and the involved tools, the content and its characteristics, and the aspects that should be preserved have to be analysed and described in a measurable way, which is time consuming and requires technical expertise. After applying the candidate solutions to sample data, measurements have to be collected, a task that can be automated if the information is drawn from tool and format registries or extracted from the results via characterisation or quality assurance tools. But to achieve this the information needs to be aligned with the specified requirements. Before a ranking of solutions can be obtained, utility functions have to be established that grade value ranges of measurements.

The collaboration between tools can partly be improved by specifying and adopting open interfaces, e.g. repositories can provide interfaces for controlled access to the content and management of preservation plans, which can be used by components specifically developed for preservation needs, like decision support and control. But there are a lot of tools, especially solution components, that can be useful but are not explicitly designed for preservation. In these cases adopting common interfaces represents a form of tight coupling which is neither desirable nor feasible.

The following sections propose an approach based on ontologies to address these problems. We adapt the planning tool and discuss its impact on the different steps of the planning workflow.

3.2 Policies and Vocabulary

In [KKP$^+$13] the authors suggest to provide a common language to achieve semantic interoperability between these components. Based on observations of other endeavours like building software and file format registries they conclude that obviously closed formats, but also strictly curated knowledge bases lead to information gaps.

This is partly due to the rapidly evolving nature of software systems, which makes it difficult to keep related information up to date. There is already a vast amount of different file formats that need to be described, and new formats are developed and existing ones are extended to reflect the advancements in tools and to capture the new feature they provide. These various versions of tools can exhibit fundamentally different behaviour and need to be tested separately, and described as well.

Moreover it prevents sharing of experiences: A tool which performs well in most circumstances might be used by an organisation in a different context or specific parameter setting, and show divergent performance - for the good or for the bad. This knowledge cannot be shared easily if the knowledge base is managed by a closed circle of responsible persons.

For these reasons a framework based on linked data principles is proposed. Instead of specifying an exhaustive model that is able to capture all required information, but would impose again its limitations on systems, a small set of vocabularies and ontologies is used to describe concepts and domain entities, express their interrelations and enable referencing existing information sources.

These are based on open standards: the Resource Description Framework (RDF)[1] is a standard model for data interchange on the web and allows to describe resources and express relationships between them as subject-predicate-object triples, forming a directed labeled graph. The Web Ontology Language (OWL)[2] is built on RDF and

[1]http://www.w3.org/RDF

[2]http://www.w3.org/TR/sparql11-overview

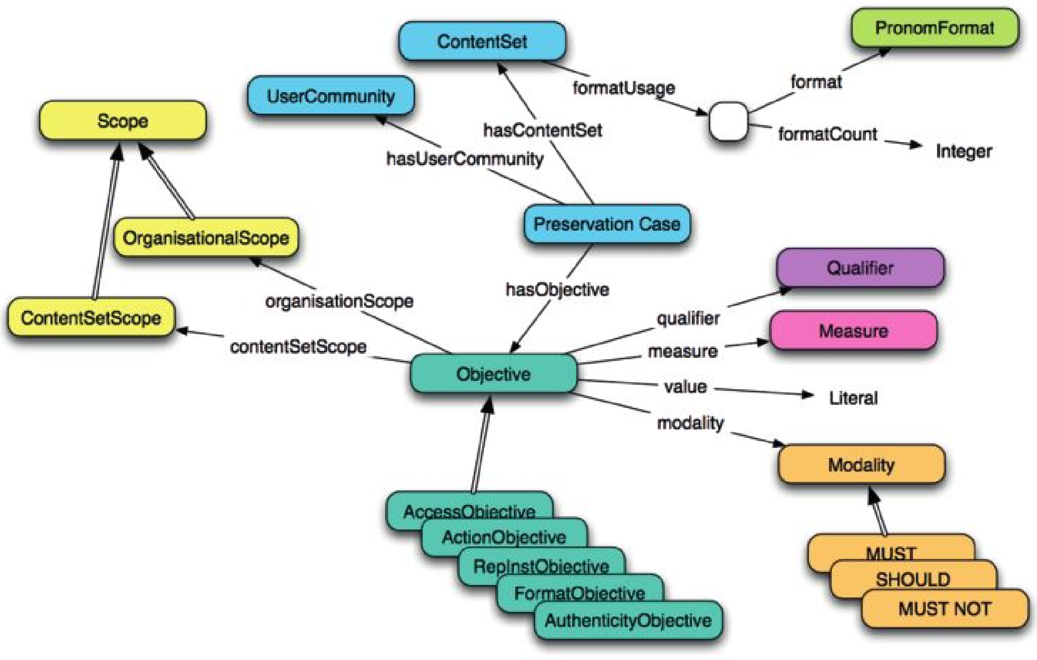


Figure 3.1: Core model of control policies [KKP+13]

provides means to describe taxonomies and manage ontologies and its formal semantics support reasoning. SPARQL Protocol and RDF Query Language[3] can be used to query or manipulate RDF graphs.

The proposed language can be used to describe solution components, which will be explained in Section 4.4, and to construe control policies in the sense of [Obj08] that are "Sufficiently detailed and precise that a person who knows the element of guidance can apply it effectively and consistently in relevant circumstances to know what behaviour is acceptable or not, or how something is understood" .

Figure 3.1 shows the core model of such control policies. Its central element is a preservation case, which describes the circumstances and the expected behaviour: It pulls together the content set which should be preserved and the user community it should be preserved for, and defines a set of objectives and constraints that should be met. These objectives and constraints reference aspects of quality of preservation actions, objects, their representations, or formats. They need to be related to measurable criteria to enable check for conformance.

A control policy is defined as a tuple of:

- a measure relating to an objective,
- a concrete value of this measure,

[3]http://www.w3.org/TR/sparql11-overview

- a qualifier (equals, less than, greater than, less or equal, greater or equal) and
- a modality that specifies how this statement should be fulfilled (must, should, must not, should not).

An example of a control policy statement is *Compression type must be none*, the corresponding RDF graph is shown in Listing 3.1 serialised as RDF/XML.

Listing 3.1: RDF representation of control policy statement *Compression type must be none*

```
<!DOCTYPE rdf:RDF [
   <!ENTITY quality "http://purl.org/DP/quality#">
   <!ENTITY measures "http://purl.org/DP/quality/measures#">
   <!ENTITY preservation-case "http://purl.org/DP/preservation-case#">
   <!ENTITY control-policy "http://purl.org/DP/control-policy#">
   <!ENTITY modalities "http://purl.org/DP/control-policy/modalities#" >
   <!ENTITY qualifiers "http://purl.org/DP/control-policy/qualifiers#" >
]>
   <owl:NamedIndividual rdf:about="CompressionTypeMustBeNone">
      <rdf:type rdf:resource="&control-policy;FormatObjective"/>
      <skos:prefLabel>Compression type must be none</skos:prefLabel>
      <control-policy:value rdf:datatype="&xsd;string">none</control-policy:
         value>
      <control-policy:measure rdf:resource="&measures;117"/>
      <control-policy:modality rdf:resource="&modalities;MUST"/>
      <preservation-case:contentSetScope rdf:resource="100yearsphotos"/>
   </owl:NamedIndividual>
```

The representation of this control policy statement is machine processable and human readable, and additional description elements increase comprehensibility.

Following the linked data principle resources are identified via URIs that should be resolvable, that is when accessed either deliver the resource itself or provide information about it, for example when they represent resources of the physical world. One problem of URIs is their stability: the referenced resources depend on web infrastructure that is subject of change. The causes for changes are manifold and may range from technical, social, or business reasons, for example the focus of a hosting organisation changes or it might be restructured internally. Anyway, relocation of resources can render existing references invalid, which is undesirable especially in the context of long term preservation.

For this reason Persistent Uniform Resource Locators (PURLs) are used as basis for identifiers of the vocabulary. The PURL service[4] provided by OCLC[5] adds an additional layer of indirection using standard mechanism of the HTTP and provides intact identifiers even if the underlying resources are relocated.

Note that a set of control policy statements cannot be used to automatically determine and adopt a preservation solution, as different goals might be competing and such conflicts need to be resolved by responsible personnel. But the following section will show how the ontologies can alleviate the involved processes.

[4]http://purl.org
[5]http://www.oclc.org

3.3 Control Policies in Plato

Plato was extended to adopt the ontology presented in the previous section to read control policies and use their information to support the planning process. Before beginning a new preservation plan the control policies have to be fed into *Plato*. The relevant control policies can be managed in *Plato*'s group settings - new control policy definitions can be uploaded, and for the active ones preservation cases and related objectives can be examined.

Jena[6], originally developed by HP Labs and adopted 2010 by the Apache Software Foundation, is an open source Semantic Web framework for Java. It supports several serialization formats for RDF, and also supports OWL. It provides a SPARQL 1.1 compliant query engine to query RDF data, and different reasoners can be integrated.

This framework is used to de-serialize the RDF graph of control policies, and preservation cases together with related information are retrieved via SPARQL queries to populate a corresponding model of Java objects. It is stored along the users group for later reference.

Figure 3.2 shows control policies of a group with two scenarios. For each of them the objectives are listed in human readable form and allow to quickly grasp the imposed constraints. The elements of the language are provided as links to the corresponding resources, this way the exact meaning of elements, for example a certain measure, can be looked up quickly.

When control policies are present, these effect the planning process in several steps:

3.3.1 Define Basis

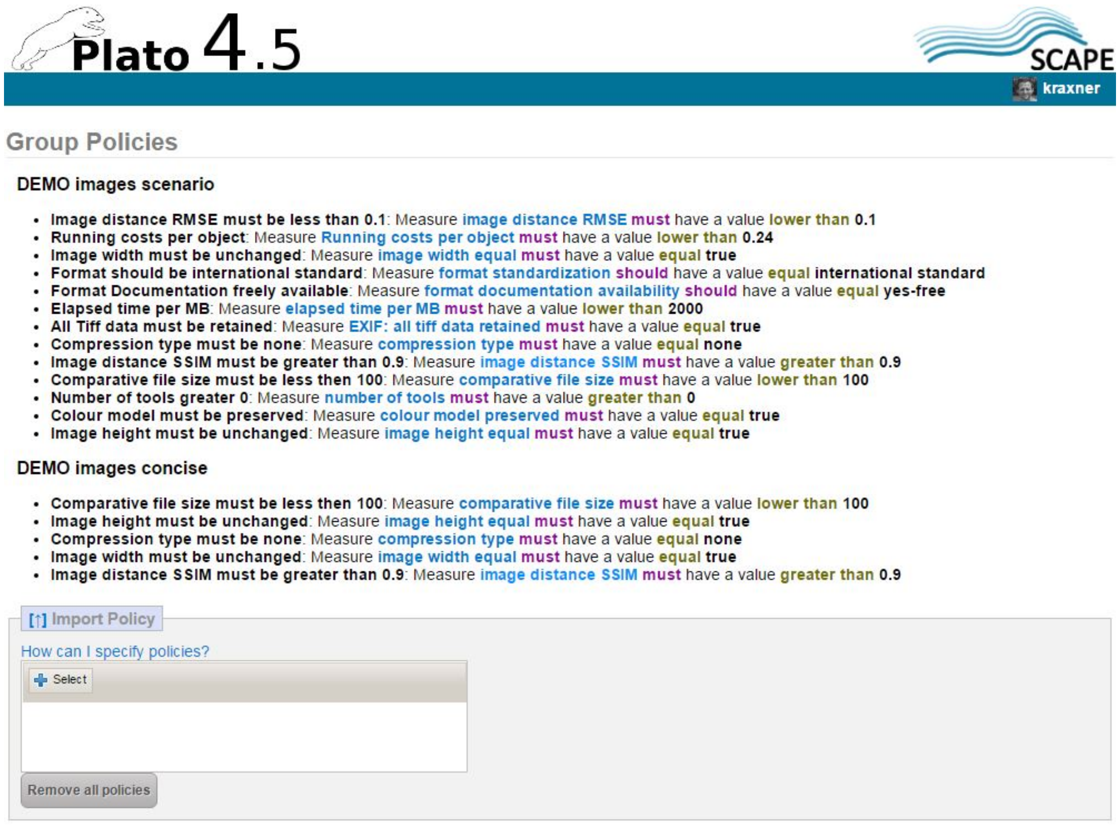
In *Define Basis* control policies provide information about the organisation and its goals: Basic information on the organisation itself is automatically applied to the plan.

A control policy definition can address multiple preservation cases, representing the scenarios of preserving content sets for different designated communities. Therefore a list of all defined preservation cases is presented to the user, who can then pick one to create a preservation plan for. The objectives of this preservation case are then stored in the preservation plan in human readable form to make this part of the ratio driving the decision process visible and keep it as evidence. Moreover, the preservation case and all related objectives are stored to be used in later steps. Figure 3.3 shows the preservation cases from the control policies loaded before, the current selection is about to be changed to *DEMO image concise*.

3.3.2 Identify Requirements

In *Identify Requirements* all requirements and constraints driving the decision process have to be documented. As discussed in Section 2.6 part of these are determined by the organisations goals and policies. Once a preservation case is selected, related objectives

[6] `http://jena.apache.org`



Group Policies

DEMO images scenario

- **Image distance RMSE must be less than 0.1**: Measure image distance RMSE must have a value lower than 0.1
- **Running costs per object**: Measure Running costs per object must have a value lower than 0.24
- **Image width must be unchanged**: Measure image width equal must have a value equal true
- **Format should be international standard**: Measure format standardization should have a value equal international standard
- **Format Documentation freely available**: Measure format documentation availability should have a value equal yes-free
- **Elapsed time per MB**: Measure elapsed time per MB must have a value lower than 2000
- **All Tiff data must be retained**: Measure EXIF: all tiff data retained must have a value equal true
- **Compression type must be none**: Measure compression type must have a value equal none
- **Image distance SSIM must be greater than 0.9**: Measure image distance SSIM must have a value greater than 0.9
- **Comparative file size must be less then 100**: Measure comparative file size must have a value lower than 100
- **Number of tools greater 0**: Measure number of tools must have a value greater than 0
- **Colour model must be preserved**: Measure colour model preserved must have a value equal true
- **Image height must be unchanged**: Measure image height equal must have a value equal true

DEMO images concise

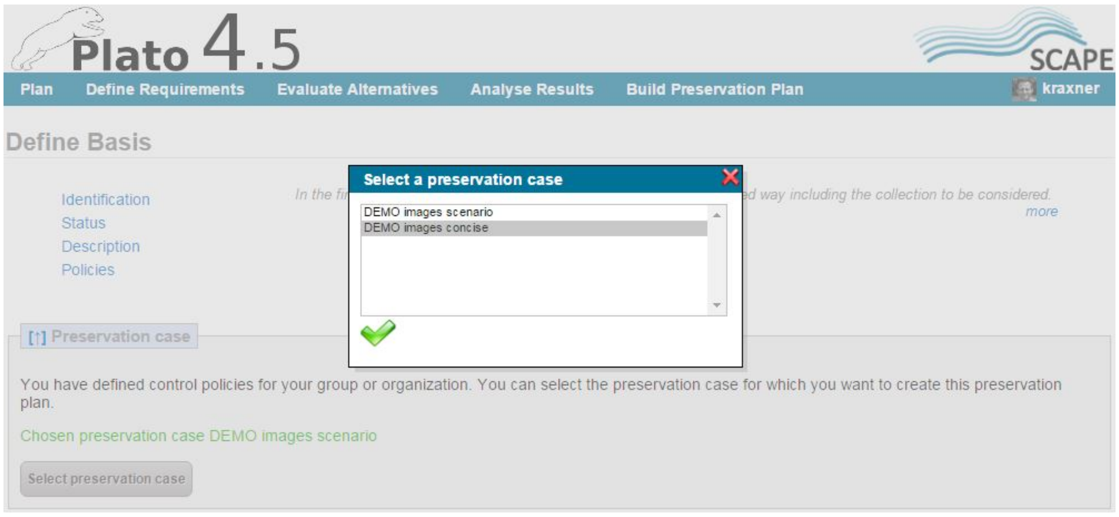
- **Comparative file size must be less then 100**: Measure comparative file size must have a value lower than 100
- **Image height must be unchanged**: Measure image height equal must have a value equal true
- **Compression type must be none**: Measure compression type must have a value equal none
- **Image width must be unchanged**: Measure image width equal must have a value equal true
- **Image distance SSIM must be greater than 0.9**: Measure image distance SSIM must have a value greater than 0.9

Figure 3.2: Screenshot of *Plato* - Control policies of two scenarios

can be used to derive the objective tree. Each objective relates to one criterion, the referenced measure provides a detailed specification of the scale, eventual restrictions, and a description of what is measured. The categorisation hierarchy of the measure catalogue can be used to structure the set of derived measures, which benefits the overview of the resulting objective tree.

The before mentioned example in Listing 3.1 would result in a tree fragment *Functional correctness: Representation Instance Property/compression/* with a criterion *compression type* of ordinal scale *none/lossless/lossy*. The criterion keeps a reference to the measure and thus enables automated evaluation.

The generation of the objective tree is automatically triggered when step *Identify Requirements* is entered and no requirements have been defined so far. Figure 3.4 shows the populated objective tree based on the previously selected preservation case *DEMO images concise*, which has a reduced number of objectives. Depending on the completeness of the control policy definition only few missing requirements have to be supplemented.



You have defined control policies for your group or organization. You can select the preservation case for which you want to create this preservation plan.

Chosen preservation case DEMO images scenario

Figure 3.3: Screenshot of *Plato* - Preservation case selection

3.3.3 Transform Measured Values

Additionally, modality, qualifier, and value defined in objectives could be used to guide the definition of utility functions for criteria in step *Transform Measured Values*. For simplification the cases of ordinal scales and numeric scales are examined separately.

Ordinal values

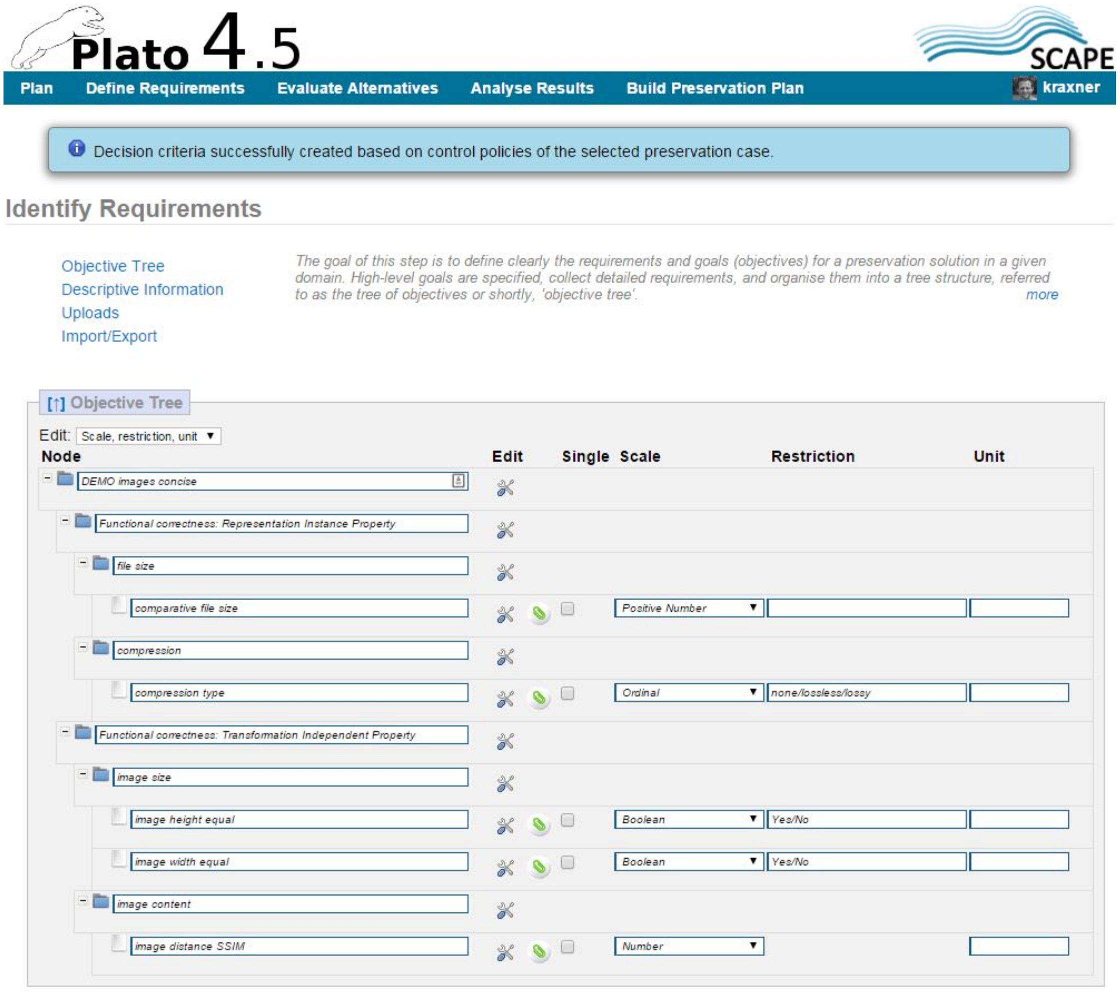
Ordinal values can only take values from a predefined finite set, enumerated in the restriction of the corresponding scale. (That is, boolean and yes/acceptable/no scales can be considered as special cases of ordinal scales.) Table 3.1 gives an overview on the utility functions derived from the modality.

A modality MUST can be translated to a utility function that assigns *max-value* to *value*, and 0 to all other possible values, marking them as not acceptable.

Modality SHOULD expresses only a preference. Clearly *max-value* can be assigned to *value*, but nothing definite is known about the target values of the remaining values, especially the intended ranking of them. They can be populated with a low target value of 1, but surely require manual review.

The utility functions for the corresponding negated modalities MUST NOT, and SHOULD NOT respectively, can be obtained by switching target values between *value* and the remaining values of the restriction.

Modality MAY is even less restrictive and only states that the given *value* is accepted. Even if such an objective does not pose strong information that can be used for automation, it can represent a notable fact valuable for documentation in the sense that the property in question has been considered.



Decision criteria successfully created based on control policies of the selected preservation case.

Identify Requirements

Objective Tree
Descriptive Information
Uploads
Import/Export

The goal of this step is to define clearly the requirements and goals (objectives) for a preservation solution in a given domain. High-level goals are specified, collect detailed requirements, and organise them into a tree structure, referred to as the tree of objectives or shortly, 'objective tree'. more

[↑] Objective Tree

Edit: Scale, restriction, unit

Node	Edit	Single	Scale	Restriction	Unit
DEMO images concise					
Functional correctness: Representation Instance Property					
file size					
comparative file size			Positive Number		
compression					
compression type			Ordinal	none/lossless/lossy	
Functional correctness: Transformation Independent Property					
image size					
image height equal			Boolean	Yes/No	
image width equal			Boolean	Yes/No	
image content					
image distance SSIM			Number		

Figure 3.4: Screenshot of *Plato* - Requirements tree generated based on control policies

Numeric values

In most cases the value range for numeric values is infinite, and even in special cases like restricted integer ranges it is likely too big to define a utility function as set of ordered pairs of input and output values. Accordingly, the methodology discussed in Section 2.2 uses thresholds for each target value starting from 1 up to *max_value* (which is set to 5 like implemented in Plato). It makes the assumption that the desired utility functions for numeric values are always monotonous, i.e. depending on the use case for a certain criterion either the higher or the lower a value the better. Therefore single discrete values are not supported and qualifier *equals* will be ignored hereafter.

But upper(U) and lower(L) bounds can be specified to restrict value ranges, and when present these can support the definition of utility functions.

The semantic of the thresholds is the following: if the threshold for target value

Modality/Utility	u(value)	u(restriction\value)
MUST	max	0
MUST NOT	0	max
SHOULD	max	1
SHOULD NOT	1	max
MAY	1	?

Table 3.1: Utility functions for ordinal values depending on modality

Modality/Threshold	1	2	3	4	5
MUST	$value + \epsilon$				U
MUST NOT	value				L
SHOULD	L	value			U
SHOULD NOT	U	value			L
MAY					

Table 3.2: Utility functions for numeric values for qualifier *greater than*

1 is not met, the value is unacceptable, otherwise the target value of the highest met threshold is achieved. Thus for increasing thresholds (with the semantics "a higher value is better") a value $<$ threshold1 is unacceptable, for decreasing thresholds a value $>$ threshold1 is unacceptable.

Table 3.2 shows the thresholds which can be deduced for qualifier *greater than*.

"Measure X MUST be greater than value" implies increasing thresholds. The specified value cannot be used as threshold1, as it should be unacceptable itself. Instead the next higher value has to be chosen as threshold - for real numbers a smallest epsilon value that distinguishes two numbers could be defined. If an upper limit was defined, it could be used as threshold5.

"Measure X MUST NOT be greater than value" on the other hand implies decreasing thresholds, and the value can directly be used for threshold1. If a lower limit was defined, it could be used as threshold5.

Modalities SHOULD and SHOULD NOT are less restrictive. It can be assumed that the specified value is on the lower end of target values, and a more extreme value should result in a higher target value. Nevertheless, it is acceptable if the value is not met, so if lower and upper limits are present they can be used as threshold1 to prevent this criterion from producing target values that lead to rejection of a preservation solution.

Table 3.3 summarises thresholds for qualifier *greater or equal than*. They can be defined like before, but for modality MUST the value can now be used directly as treshold1, and in turn for MUST NOT an epsilon has to be considered.

Some notes on the proposed method: Utility functions for the remaining qualifiers can be derived using negated equivalents of the ones above. Where both, threshold1 and threshold5, are present, missing thresholds could be interpolated.

Modality	1	2	3	4	5
MUST	value				U
MUST NOT	$value - \epsilon$				L
SHOULD	L	value			U
SHOULD NOT	U	value			L
MAY					

Table 3.3: Utility functions for numeric values for qualifier *greater or equal than*

If the suggested method of using lower and upper limits of ranges as thresholds is appropriate depends very likely on how the specified range relates to the actual measurements, and requires further validation.

3.4 Summary

In this chapter first an ontology was identified which can be used to describe policies in a machine processable way. Then it was described how *Plato* was adapted and how some steps of the preservation workflow benefit directly from this integration: Context information from control policies is applied to the plan in step *Define Basis*, and in step *Identify Requirements* a tree of decision criteria can be derived from their objectives. Finally the potential to derive utility functions from policies was discussed.

The next chapters outline how planning can be integrated with related systems and support monitoring of quality. Specifically, Section 4.4 will show how the adopted vocabulary is used to describe workflows which enables automatic lookup and composition of quality assurance workflows and to combine them with migration actions to executable plans. And Section 5.4 discusses how monitoring conditions can be generated for *Scout* and to check outputs of annotated workflows when they also refer to the presented vocabulary. A bigger picture of the role of these ontologies for preservation systems is given in Chapter 6, where results are evaluated based on a case study.

CHAPTER 4

Systems Integration

This chapter discusses how advances in content profiling, repository APIs, and workflows and operations can be utilized for preservation planning.

It starts with integration of *Plato* with *c3po* to facilite describing the collection and sample selection. Existing repository APIs are integrated to access content, and APIs to manage preservation plans require schema modelling and extension of existing schemas.

Not part of this work was the annotation of workflows and definition of a taxonomy which allows automatic composition. Nevertheless it is discussed briefly, as it is a result of making *Plato* ontology aware as described in the previous chapter, and basis for improved automated evaluation and complete specification of an executable preservation plan.

Finally, new APIs are developed to integrate *Plato* with *Scout* and thus improve continuous monitoring.

4.1 Integration with c3po

As mentioned earlier, part of the planning workflow is to gain insight into the content that should be preserved, describing it, and identifying representative sample objects, which is challenging due to content's volume and complexity. A representative set should cover all essential characteristics of the content. However, each candidate solution has to be tested with every sample, and for each pair all criteria related to the outcome have to be evaluated, hence related effort increases rapidly. Therefore the representative set should be as small as possible. Selected objects depend highly on the used algorithm, consequently this is important evidence and should be documented.

Currently *Plato* only allows to document this information, but leaves analysis to the user, who has to enter all findings manually afterwards. Beside uploaded bit-streams and their characterisation information the data is unstructured and therefore hinders automation.

The content profiling tool *c3po* presented in Section 2.3 aggregates characterisation data of a digital object repository's content, and provides a GUI to interactively explore the data, analyse its features, and drill down to a subset of interest, for which representative sample objects can be determined.

While this helps the user to gain insight into the content, *c3po* was built also with preservation planning in mind. The possibility to export the set of identified sample objects was added, which results in a so called *content profile*. Listing 4.1 shows a shortened example of such a profile, the complete document can be found in Appendix A.2.

A content profile contains information about the collection it is about, its name and the total number of objects, and the filters that were applied to determine the subset of interest, the *partition*. For the latter a summary of the characteristics of the corresponding objects is given as histogram, for example for feature *compression_scheme* the encountered values together with their occurrences. The algorithm used to determine the sample objects is given, together with the list of samples, their unique identifier and concrete characterisation information. Finally, all elements of the partition are identified, in this example by listing their unique identifiers.

Listing 4.1: Summarised *c3po* content profile

```xml
<profile xmlns="http://ifs.tuwien.ac.at/dp/c3po" xmlns:xsi="http://www.w3.org
    /2001/XMLSchema-instance" collection="demo" date="Wed Jul 23 09:05:29
    WEST 2014" count="206">
  <partition count="104">
    <filter id="61d29510-9031-41be-8f21-6cac7192424f">
      <parameters>
        <parameter>
          <name>collection</name>
          <value>demo</value>
        </parameter>
        <parameter>
          <name>metadata.mimetype.value</name>
          <value>image/tiff</value>
        </parameter>
      </parameters>
    </filter>
    <properties>
      <property id="format" type="STRING" count="104">
        <item id="Tagged Image File Format" value="104"/>
      </property>
      <property id="compression_scheme" type="STRING" count="104">
        <item id="LZW" value="102"/>
        <item id="Uncompressed" value="2"/>
      </property>
    </properties>
    <samples type="size'o'matic 3000">
      <sample uid="roda:71/R2013-03-22T17.20.40.45Z/F6">
        <record name="puid" value="fmt/7" tool="Droid 3.0"/>
        <record name="compression_scheme" value="LZW" tool="Exiftool 9.06"/>
        <record name="image_width" value="4672" tool="Exiftool 9.06"/>
      </sample>
    </samples>
    <elements>
      <element uid="roda:79/R2013-03-22T17.17.51.70Z/F7"/>
      <element uid="roda:79/R2013-03-22T17.17.51.70Z/F11"/>
    </elements>
  </partition>
</profile>
```

This information is valuable for the planning process, therefore *Plato* was extended to read these profiles: In step *Define Sample Objects* a collection profile can be uploaded. The information about the collection is applied automatically, and for each sample in the profile a corresponding sample object in *Plato* is added, including its characterisation information. Figure 4.1 shows *Plato* in the step *Define Sample Objects* after loading the simplified content profile with one sample object from above.

Evenmore, if *Plato* is connected to the repository that manages this content, the samples themselves are retrieved automatically, so experiments can be conducted later on actual data. This will be explained further down in Section 4.2.2.

Both, the information on the profile and the UIDs of the elements, can later be used for the Preservation Action Plan to identify the objects the preservation action should be applied to, see Section 4.2.3.

Thus, the planning workflow is integrated directly with *c3po* by the means of content profiles, so that the technical processes of content characterization and selection of sample data for experimentation are fully automated, and the analytical step of analysing content sets is well supported.

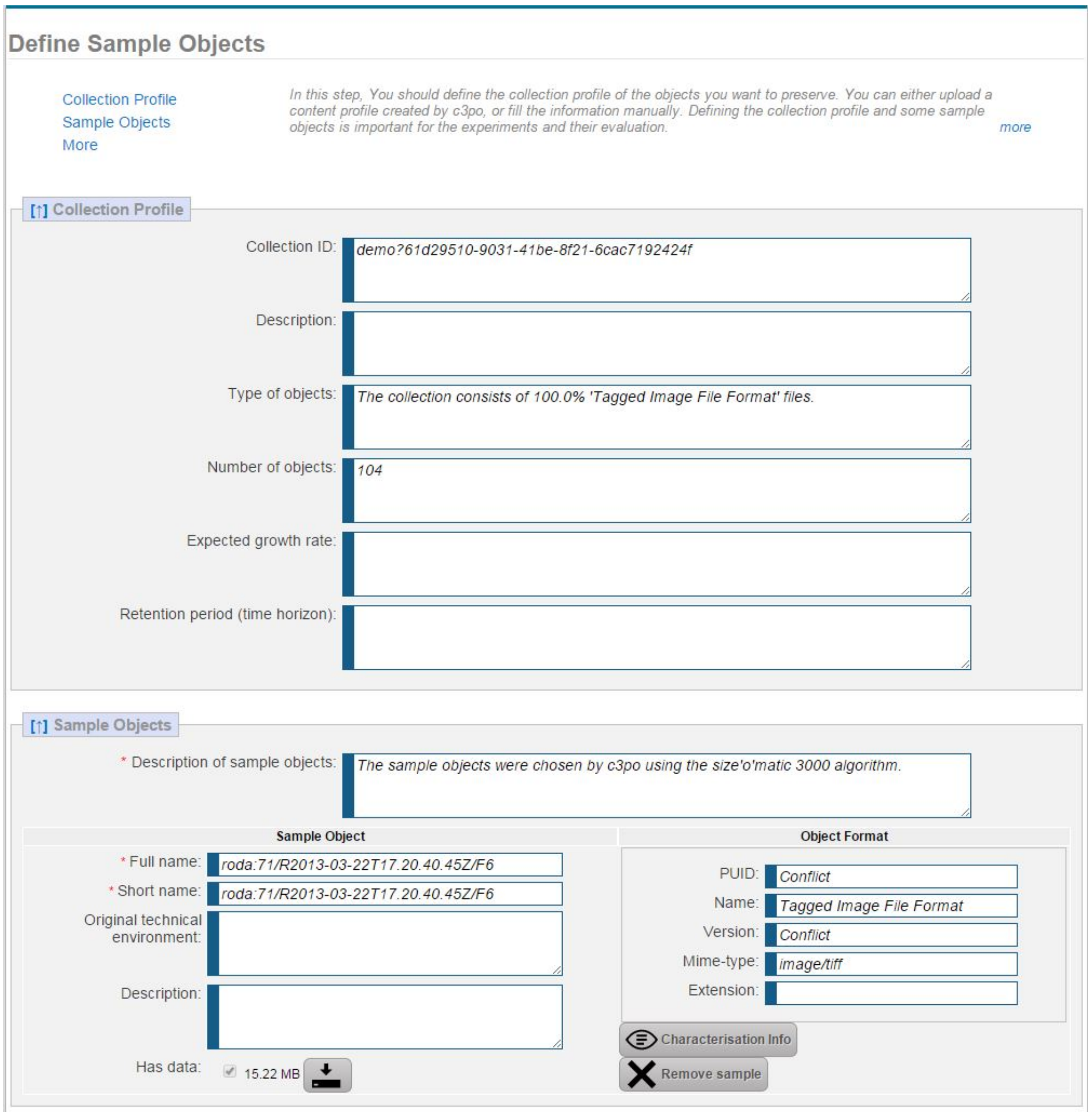
4.2 Repository

In Section 2.5.1 the role of repositories for managing digital objects is explained, and how SCAPE defines open interfaces to manipulate digital objects and their meta data, and to manage preservation plans.

The previous section discussed how content profiling tools can provide concise and structured information on the content and representative samples. But the user still needs to manually retrieve the objects from the repository, and to upload them to *Plato*, if experiments should be conducted automatically. This requires additional work and is error prone, an automated procedure can improve confidence in the procedure.

The final preservation plan contains all information to adopt the chosen strategy in the repository and apply it to all concerned objects, but so far it serves only as documentation. An early attempt was tailored to one specific repository, working only with migration actions from one registry, and required to complete the ad hoc actionable plan manually. But a sustainable solution needs to provide repository independent interfaces to submit preservation plans and manipulate them, and has to define a concise specification of the information required to execute a plan, which should happen in an automated way to make sure that the components are employed as tested before.

While *Plato* can be used as standalone tool to create preservation plans for a set of objects, the process can benefit a lot from an integration with such a repository. The following sections describe the necessary additions.


Define Sample Objects

Collection Profile
Sample Objects
More

In this step, You should define the collection profile of the objects you want to preserve. You can either upload a content profile created by c3po, or fill the information manually. Defining the collection profile and some sample objects is important for the experiments and their evaluation. *more*

[↑] Collection Profile

Collection ID:	*demo?61d29510-9031-41be-8f21-6cac7192424f*
Description:	
Type of objects:	*The collection consists of 100.0% 'Tagged Image File Format' files.*
Number of objects:	*104*
Expected growth rate:	
Retention period (time horizon):	

[↑] Sample Objects

* Description of sample objects: *The sample objects were chosen by c3po using the size'o'matic 3000 algorithm.*

Sample Object

* Full name:	*roda:71/R2013-03-22T17.20.40.45Z/F6*
* Short name:	*roda:71/R2013-03-22T17.20.40.45Z/F6*
Original technical environment:	
Description:	
Has data:	✓ 15.22 MB

Object Format

PUID:	*Conflict*
Name:	*Tagged Image File Format*
Version:	*Conflict*
Mime-type:	*image/tiff*
Extension:	

Figure 4.1: Screenshot of *Plato* - Define Sample Objects via content profile

4.2.1 Connecting to a Repository

The SCAPE architecture aims to define a preservation environment of loosely coupled systems, using light weight protocols for interactions. In this sense the defined APIs specify REST interfaces for these services, and employ HTTP basic authentication for simple access restriction.

Hence, as a prerequisite to connect *Plato* with such a repository, the repository address and credentials have to be configured. To allow different organisations to use their own repositories, the group settings for *Plato* users have been extended with the repository configuration, where the URL to the repository endpoint can be stored.

While the username is also stored in the groups setting, the password is requested on demand as the planning process involves tasks requiring expertise in different fields and as such is inherently collaborative. Once this is set up, this connection can be used in different steps.

4.2.2 Sample Selection

Section 2.3 explained the importance of selecting representative sample objects to conduct meaningful comparison of the available preservation actions and to obtain insightful results. This task need not to be left to the planner. As discussed in Section 4.1 tools like *c3po* help to analyse the content and its meta data, and their output can support this task and automate it to a great extent. In case *c3po* was run on meta data from a SCAPE digital object repository, the resulting content profile also includes the unique identifiers of samples and elements in the repository. These can be used to retrieve the binary representations. The Data Connector API does provide services via HTTP to manipulate the content. Beside ingest and retrieval of intellectual entities and their meta data, also single files can be fetched.

We implemented a REST client for *Plato* to access digital object repositories. It is used to retrieve the sample objects identified by *c3po*: When the collection profile is uploaded to *Plato*, each sample object is retrieved automatically via the REST client and added to the preservation plan together with its characterisation information. This is already visible in Figure 4.1, which shows one sample object with present binary data. These sample objects are then available to conduct automated experiments and evaluation of the preservation actions.

Hence, *c3po* can be used to analyse the content as *Plato* understands content profiles, and once it is connected to a repository that supports the Data Connector API step *Define Sample Objects* is automated.

4.2.3 Plan Deployment

After the best preservation action has been identified and the preservation plan has been validated, the chosen strategy should be applied to all related objects in the repository. To maintain trustworthiness the repository needs to keep track of all changes to its content. The rationale behind the applied changes is valuable information, therefore the provenance information of the object should be updated accordingly. Because of this, it is necessary to store the complete plan in the repository, so it can be referenced by affected objects.

At the same time scalability of operations has to be considered: A finished preservation plan contains all evidence which led to the final decision, including sample objects and migration results, which can add up to a large amount of data. Additionally, XML serialisation of the plan requires binary data to be Base64 encoded, inflating the resulting file even more. Therefore it is not desirable to pass the complete plan on to the execution engine.

Plan Management API

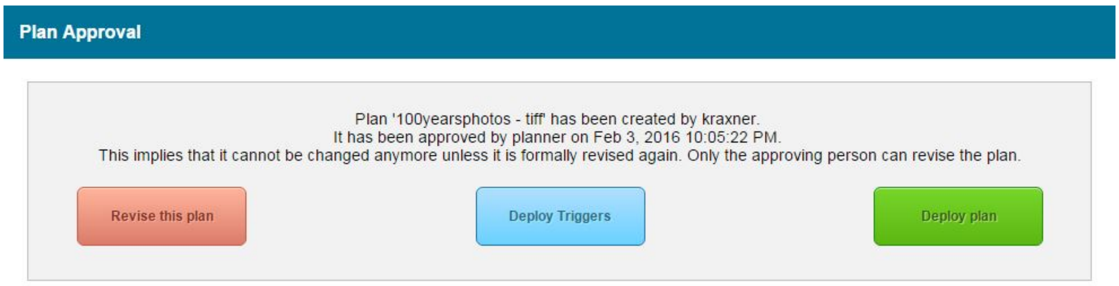
From creating a preservation plan to implementing it in the repository a number of components are involved. In the SCAPE environment these are:

- Planning Component: Used to create new plans or revise existing ones
- Digital Object Repository: Responsible to store and preserve objects and finished plans
- Plan Management Application: Allows to schedule plans for execution and to monitor active plans.
- Plan Execution Environment: Applies preservation action to objects in the repository.

These components need to collaborate efficiently to fulfil their role in the preservation environment. To this extent the lifecycle of a preservation plan had to be analysed and the requirements for each component specified. This task was led by a project partner and resulted in the definition of the Plan Management API [1]. It describes the interfaces as a set of HTTP endpoints, the responsibilities of clients and servers and the data exchanged. (The latter required some additions which will be described in the following section.)

For the planning component two endpoints are of special interest:

- Retrieve a unique identifier for a plan: Before a plan is deployed to the repository, a unique identifier is retrieved and stored in the plan. This can be used later to look up the plan, e.g. when the monitoring tool *Scout* requests a re-evaluation of results.
- Deploy a plan: A validated plan can be deployed directly to the repository which was configured earlier in the user's group settings.


Plan Approval

Plan '100yearsphotos - tiff' has been created by kraxner.
It has been approved by planner on Feb 3, 2016 10:05:22 PM.
This implies that it cannot be changed anymore unless it is formally revised again. Only the approving person can revise the plan.

Figure 4.2: Screenshot of *Plato* - Deploy plan to repository, and triggers to *Scout*

[1]https://github.com/openpreserve/scape-apis/blob/master/PlanManagementAPI-FINAL-1.1.pdf

We have implemented and integrated REST clients for these endpoints in *Plato*. When a finished plan has been approved, an option is provided to deploy it directly to the repository, as shown in Figure 4.2. This retrieves a unique identifier and stores it in the plan, which subsequently is pushed to the repository and is then available for execution.

Preservation Action Plan

For adopting a chosen strategy and applying it to all related objects in the repository a subset of the preservation plan is sufficient. The *Preservation Action Plan* has been specified as a container for more efficient data exchange and allows more concise interfaces for components involved in execution of plans. It contains all the information necessary to apply the preservation action to the objects in the repository:

- Collection: Identifier and name of the collection the objects are part of.
- Objects: The plan identifies the objects that the preservation action should be applied to. These might be identified by enumeration of object identifiers, or an SRU[2] query that selects a subset of the collection. This information stems from the content profile generated by *c3po*, and can be used to request the content from a Data Connector API implementation.
- Executable Plan: The Executable Plan applies the preservation action to an object and provides measures used to monitor the quality of the action. At the moment this is a Taverna workflow that conforms to the *Executable Plan* Component Profile.
- Quality Level Descriptions: These give expected levels of quality and are derived from the decision criteria and their utility functions. They are encoded as Schematron schemas and can be used to validate the quality measures produced by the Executable Plan. See Section 5 for a detailed description on how QLDs are generated and evaluated.

We defined an XML Schema (see Appendix A.1) to complete the specification of interfaces involved in plan execution. It is also included in the schema of the preservation plan, to store the Preservation Action Plan as evidence.

4.3 Scout

In course of the preservation planning workflow a plan is created, and when finally approved, the changes discussed in the previous section support automated deployment and execution. In [KPD+13] the SCAPE Planning and Watch Suite is presented, which already demonstrates the benefits and opportunities of a planning component that considers the key components discussed so far. However, the results of preservation plans in action are not monitored yet, thus no quality assurance is in place that ensures that

[2] http://www.loc.gov/standards/sru/

the requirements identified during the planning process are also met during operations. Section 5 will address this problem, and discuss how the adopted quality model can be used to generate triggers for *Scout*.

When such a monitoring event is triggered, the cause of the reported changes has to be investigated, and may require further preservation planning activities, e.g. the creation of a new preservation plan or re-evaluation of an existing one. Currently planners are informed of new risks or opportunities only via email, by triggers which had to be created manually in *Scout*.

In this section a new API is specified which allows a closer integration with monitoring systems, to improve feedback on changes in the environment.

4.3.1 The Notification and Assessment API

The Notification and Assessment API is a set of HTTP endpoints to integrate *Plato* with monitoring systems. It allows to notify the planning component of potential risks or opportunities, and to ask for re-assessment of a preservation plan when one of the influencing factors has changed.
There are two main use cases: Notifying a planner about new risks or opportunities, and re-assessment of a plan. Both endpoints use HTTP basic authentication to restrict access to this services.

Notification

A planner can store watch triggers in *Scout* to be notified about changes in the environment. This can be done manually with the web interface of *Scout*, and with the changes described in Section 5.5 *Plato* can automatically generate triggers. *Scout* can deliver notifications via email, but so far they consist of the trigger definition, which is hard to understand. They should be also visible in the planning component, and more specific information could support the user in resuming the planning process.

A Notification can consist of following properties shown in Table 4.1:

message	The message which is shown to the user
plannerEmail	The email address to identify the planner who should be notified
planId	The repository identifier of the preservation plan the notification relates to. If provided all related users are notified.
measureUri	The URI of the measure the notification relates to, in case a trigger was fired.(optional)
value	The actual value of the measurement, which exceeded predefined boundaries. (optional)

Table 4.1: Notification API - Properties of a notification

A REST endpoint that allows agents to store new notifications is implemented in *Plato*. Its specification is given in Table 4.2.

Name	Store notification
Path	/notification
Method	HTTP/1.1 PUT
content type	application/json

Consumes	Notification: e.g.:
	{
	message : [string],
	plannerEmail : [string],
	planId : [string],
	measureUri : [uri],
	value : [string]
	}

Success response	Code: 201 Created
Error response	Code: 401 Unauthorized

Table 4.2: Notification API - Specification

Reassessment of Plans

In the course of the preservation planning workflow a planner defines all criteria together with their utility functions which influence the decision to adopt a certain preservation action. If they are used to generate triggers, *Scout* can monitor if measurements violate predefined mandatory ranges for some criteria. But it cannot evaluate the impact of minor changes on the overall outcome of the evaluation, e.g. when measurements exceed suggested ranges for some criteria.

Instead of demanding user interaction to determine potential effects, the planning component can provide the mean to re-assess a plan with respect to changed measurements.

To this end a REST endpoint to request a re-assessment is defined, its specification is given in Table 4.3. The specification for both endpoints is publicly available on GitHub[3].

Picture a scenario in which two migration actions A and B are evaluated, and one requirement is that these should be reasonable fast, do not take more than X seconds. Migration action A is identified as the most suitable solution, the plan is deployed to the repository and executed, and triggers are generated and deployed to the monitoring component. If for some objects action A needs longer than X seconds, the trigger is activated. At the moment a notification is issued to inform the responsible planner, who can be pointed to the corresponding plan when accessing *Plato*.

To reduce human interaction the monitoring component could first query the re-assessment endpoint of *Plato* to re-calculate the ranking of candidates with the new value. Only in case the order changes a warning would be issued that further action is necessary and the plan has to be revised.

[3]https://github.com/openpreserve/scape-apis/blob/master/Notification_and_Assessment-API-V0.1.pdf

Name	Re-assess evaluation result
Path	/assessment/<planid>?alt=<alt-name>&m=<measure Id>&value=<currentval>
content type	application/json
Method	HTTP/1.1 GET
Parameters:	
planId	The repository identifier of the preservation plan.
alt-name	The name of the alternative the measurement relates to
measure Id	the ID of the measure the measurement relates to, e.g for http://purl.org/DP/quality/measures#98 this would be 98.
currentval	The current measurement which requires a re-assessment of the plan.
Success response	Code: 200 Ok The name of the highest ranked alternative if it differs from the previously selected alternative, otherwise an empty string.
Error response	Code: 401 Unauthorized

Table 4.3: Re-assessment API - Specification

4.4 Workflows as Preservation Components

Tools used for migration of objects and quality assurance of results are diverse. They have specific requirements on the runtime environment, and provide usually unstructured output and non standardised interfaces. Section 2.5.3 already mentioned the use of workflows to describe employed tools and their parameter settings, and how they are composed, but the output of the workflow itself is similarly unstructured.

The authors of [KKP$^+$13] propose to define preservation components, for which they identify the following requirements:

- **Publishing** and **Discovery** to enable sharing of solution components and expertise and reuse of components during planning
- **Automated execution** during planning to reduce effort of experiments, to increase scalability of operations, and to include quality assurance of results
- **Reproducibility** of results, as a key requirement for evidence based decision making and trustworthiness.
- **Standardised output** with well defined interfaces enables automated evaluation during planning, and allows to compare results
- **Composition** of components is required to ease the construction of new preservation components during planning, and automate the creation of executable plans that can be run in repositories.

Taverna workflows as a whole and specific parts of a workflow can be semantically annotated with RDF statements, and myExperiment provides an endpoint to query for workflows with specific annotations. With sight to this [KKP+13] suggests an ontology to describe preservation tools[4] based on the vocabulary presented earlier.

Figure 4.3 gives an overview of its elements. Following from right to left, workflows can invoke external tools to process data. These might have dependencies which need to be satisfied, a package manager configuration enables automated installation procedures, and they require an installation to be run in a certain environment.

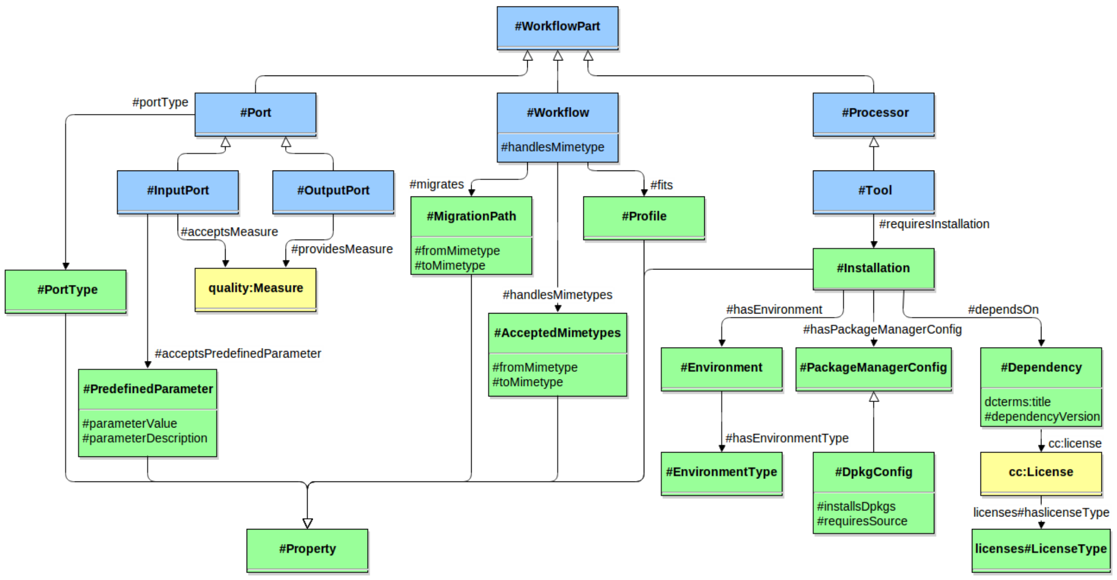


Figure 4.3: Overview of Components Ontology[KKP+13]

Three main types of components are identified: migration components that transform one object from one format to another, characterisation components that extract measurements from an object, and quality assurance tools that evaluate authenticity and validity of objects, by either comparing two objects directly for similarity, comparing pairs of measurements, or examine an object with respect to some measures. Each of these component types require a specific set of input and output ports. These sets are described as component profiles and allow to validate if a workflow is annotated appropriately, facilitate lookup of specific component types and ease automatic composition. To enable this a workflow has to declare to which profile it adheres to. Workflows can specify accepted MIME types, for migration workflows the migration path can be defined.

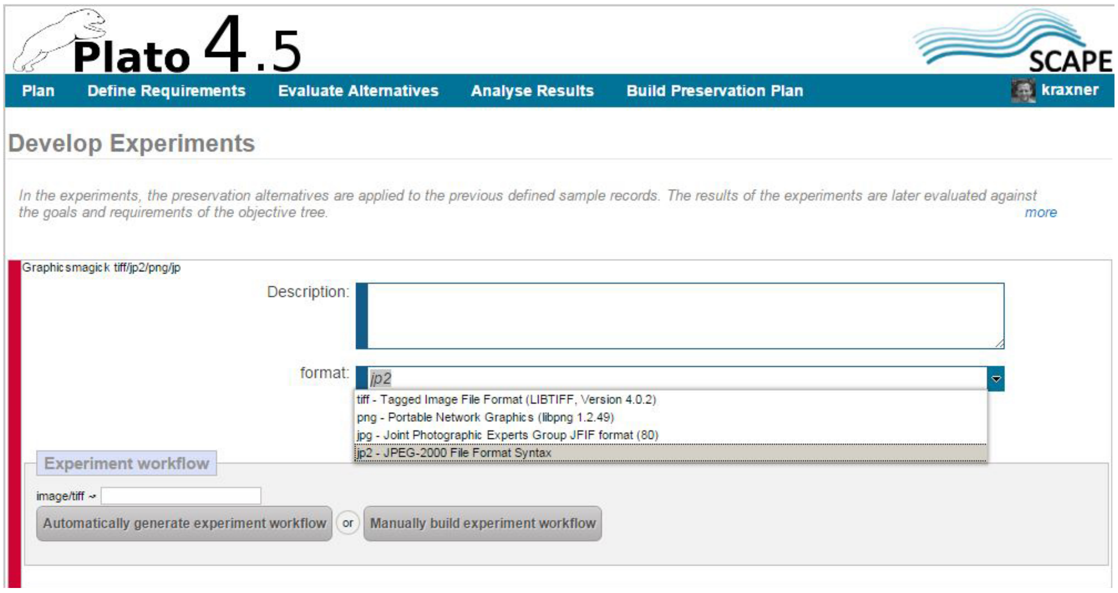
Input and output ports can be marked as such. Further it can be defined if they consume or produce specific measures, or if a input port accepts parameters. In the latter case predefined parameter sets can be provided.

[4] `http://purl.org/DP/components`

4.4.1 Integration in Plato

Note that specifying component profiles and the changes described in this section - integration of myExperiment, automatic composition of components, and execution of Taverna workflows for experimentation - are not part of this work but were done by others. Nevertheless, they play an important role: They were enabled by adapting *Plato* to the vocabulary, and on the other hand increase automation and are an integral part of the preservation action plan.

Plato was integrated with myExperiment to lookup components. The user interface allows to filter migration components according to accepted mime types, and environments they are available for. During experiment setup preconfigured parameter settings are presented to the planner, Figure 4.4 shows a component where the target format can be chosen.



Develop Experiments

In the experiments, the preservation alternatives are applied to the previous defined sample records. The results of the experiments are later evaluated against the goals and requirements of the objective tree. more

Graphicsmagick tiff/jp2/png/jp

Description:

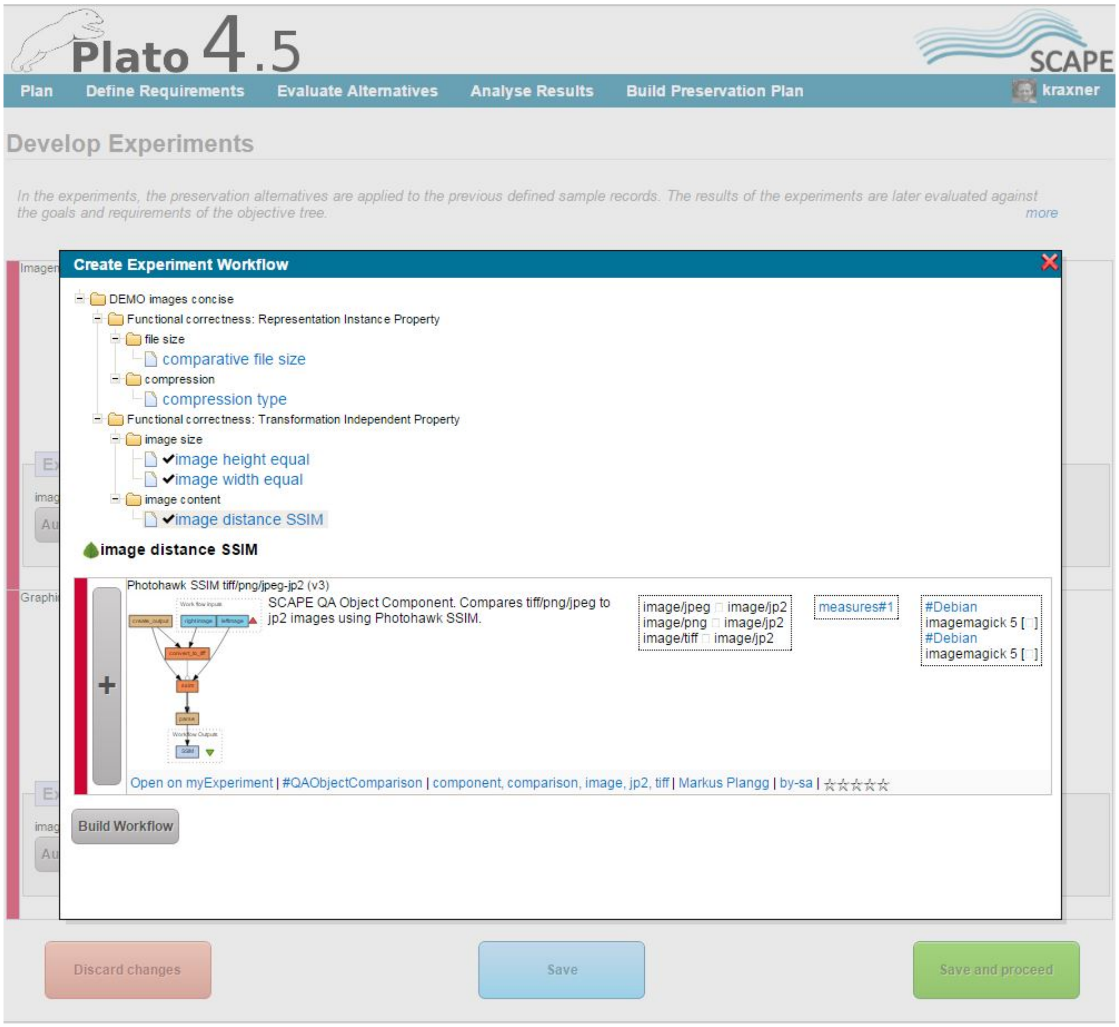
format: jp2

tiff - Tagged Image File Format (LIBTIFF, Version 4.0.2)
png - Portable Network Graphics (libpng 1.2.49)
jpg - Joint Photographic Experts Group JFIF format (80)
jp2 - JPEG-2000 File Format Syntax

image/tiff →

Figure 4.4: Screenshot of *Plato* - Develop experiments with preconfigured parameter settings

Semantic annotations of input and output ports of workflows ease composition: Quality assurance components can be automatically looked up according to the criteria defined before, and composed with migration components to fully functional evaluation workflows. Figure 4.5 shows how available quality assurance components can be queried and assigned to criteria from the objective tree. The evaluation workflows are can be run automatically on a dedicated Taverna server, and results of annotated outputs can be automatically applied to the plan.



Develop Experiments

In the experiments, the preservation alternatives are applied to the previous defined sample records. The results of the experiments are later evaluated against the goals and requirements of the objective tree. more

Create Experiment Workflow

- DEMO images concise
  - Functional correctness: Representation Instance Property
    - file size
      - comparative file size
    - compression
      - compression type
  - Functional correctness: Transformation Independent Property
    - image size
      - ✓image height equal
      - ✓image width equal
    - image content
      - ✓image distance SSIM

image distance SSIM

Photohawk SSIM tiff/png/jpeg-jp2 (v3)

SCAPE QA Object Component. Compares tiff/png/jpeg to jp2 images using Photohawk SSIM.

image/jpeg image/jp2
image/png image/jp2
image/tiff image/jp2

measures#1

#Debian imagemagick 5 [ ]
#Debian imagemagick 5 [ ]

Open on myExperiment | #QAObjectComparison | component, comparison, image, jp2, tiff | Markus Plangg | by-sa | ☆☆☆☆☆

Figure 4.5: Screenshot of *Plato* - Composing a workflow with integrated quality assurance

4.5 Summary

The previous sections discussed the adaptation of *Plato*'s architecture to integrate with adjacent systems. The component diagram shown in Figure 4.6 gives an overview of involved systems and interfaces, contributions of this work are highlighted in bold.

c3po is integrated via its content profiles. They contain aggregated information on the content set, representative samples, and a possibility to identify the objects of the content set. *Plato* can use the Data Connector API to retrieve sample data from the repository, using the identifiers from the content profile. Migration and characterisation components matching the specified criteria can be looked up on MyExperiment, and automatically composed to an Executable plan. A preservation action plan is generated and stored in the preservation plan. This can be deployed to the repository via the Plan

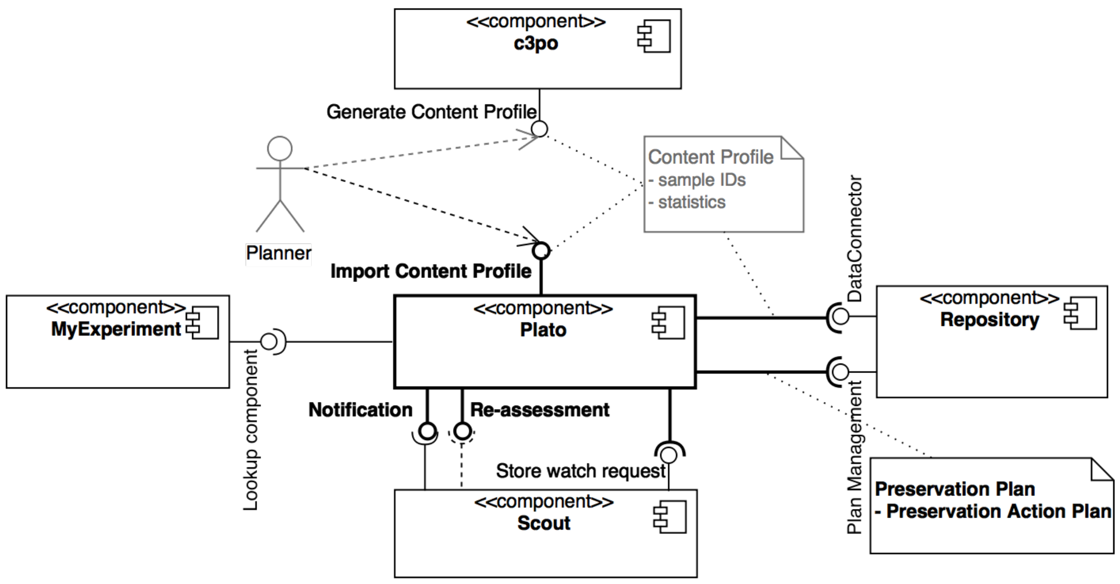


Figure 4.6: Integration of *Plato* with components of a preservation system

Management API, and corresponding watch requests are stored in *Scout.* When a watch request raises an event, *Scout* can notify *Plato* via the Notification API, or could trigger a re-evaluation via Reassessment API, once implemented by *Scout.*

The following chapter discusses how monitoring conditions can be derived from a preservation plan automatically, before a case study is presented to evaluate the results.

CHAPTER 5

Quality Assurance and Monitoring

This chapter discusses how preservation planning can support monitoring of operations, involving *Scout* and operational deployment of preservation actions.

First the requirements on a monitoring solution for preservation solutions are analysed. A methodology to derive statements on quality from decision criteria and related utility functions defined during planning is presented. Then it is discussed how monitoring conditions for *Scout* and operations can be defined.

5.1 Requirements on Monitoring Active Preservation Solutions

In Section 2.2 the preservation planning workflow and its goal to determine the best suitable preservation action to preserve content was described. During the process a set of decision criteria is defined, which provides the basis for an objective comparison of alternatives. Experiments are conducted and results evaluated with respect to these criteria, providing evidence to support the decision for a solution. In the end this solution is adopted in the repository, and in case of a migration component applied to all objects targeted by the plan. Naturally, it is expected that this tool continues to perform similar than during experimentation on the sample objects. To ensure this, ongoing monitoring of results is necessary.

However, by continuously checking if the results conform to the ones during evaluation only part of the criteria influencing the ranking are covered. New formats with better fitting characteristics might be specified, or a format itself could change for example its applying license model. Similarly, existing tools could be updated to resolve problems, or new tools with better performance could be created. Finally, requirements of the designated community could change. Therefore the environment also needs to be watched to capture new user trends and technologies.

The need for monitoring operations and environment requires a set of concepts and systems to be in place, the following monitoring requirements(MR) have been identified:

MR.1 means to identify and describe aspects of quality

MR.2 means to lookup relevant information or tools to satisfy information need

MR.3 means to express expected performance and levels of quality

MR.4 environment to collect measures, and check for conformance

MR.5 consider scalability of conformance checks

The vocabulary discussed in Chapter 3 provides a catalogue for attributes and measures[1] to describe aspects of quality. It is based on open standards and flexible and extensible, and suitable to satisfy MR.1.

The preservation watch system *Scout* presented in Section 2.5.2 aggregates information on the environment and stores it in its knowledge base. It allows to define queries which are periodically executed to trigger events when specified conditions are met, and is based on Linked Data principles and therefore can be integrated with the vocabulary. Migration results have to be examined with the help of characterisation and quality assurance tools. Section 4.4 explained how workflows can be used to wrap these tools, and the presented ontologies can be used to describe them to enable specific lookup and automated composition. Both, *Scout* and annotated workflows, can provide information to monitor, thus satisfy MR.2.

5.2 Specification of Expected Quality

To examine how expected levels of quality can be expressed a closer look at the evaluation process during planning is necessary. There, a set of decision criteria is defined to capture the important aspects of the data to be preserved, the environment, and of the actions to be applied. Based on these criteria, preservation actions in question are evaluated and a ranking is calculated. These can be used to specify monitoring conditions.

5.2.1 Transformers

Depending on the measure, the measurements results in values of different scales. Before these values can be aggregated to calculate the score of an action, utility functions have to be defined to transform them to a uniform target scale of [0,1..5] or [1,2..5] respectively.

A value of 0 will set to overall score of the action to 0, if multiplicative aggregation is applied, thus render this action unacceptable.

[1] http://purl.org/DP/quality

Criteria	Upper	Mid	Lower
C1	5 (YES)		0 (NO)
C2	5 (YES)		3 (NO)
C3	5 (>10)	1 (>4)	0 (< 3)

Table 5.1: Transformers for criteria C1-C3

Plan	C1	C2	C3	Score+	Score*
A1	YES	YES	12	16	>0
A2	**NO**	NO	**2**	1	0
A3	YES	YES	7	11	>0

Table 5.2: Ranking of three alternatives based on measurements for criteria

- if the target scale of a criterion includes 0, this means that this measure MUST (/NOT) have a certain value.
- Likewise a target scale from 1-5 could be read as a suggestion: this measure should have a certain value (the closer to 5 the better)

Consider the example given in Table 5.1. It shows three criteria with their utility functions. The latter are defined by providing selectors (in brackets) for lower, middle, and upper target values. The first two criteria are of boolean scale, and in both cases a *YES* value is optimal. Yet, for C1 *NO* is a not acceptable value, while for C2 it results only in a low target value. The third criterion has a numeric scale, and thresholds are given for each target value. C3 again has the potential to render an action unacceptable if its measurement is less than 3.

5.2.2 Ranking of Alternatives

Continuing the example, these criteria are part of a plan where three alternatives (A1, A2, A3) have to be compared and experiments have already been conducted. Table 5.2 shows resulting measurements and how alternatives can be ranked based on the transformed values: A2 will be rejected due to not acceptable values for C1 and C3 (in bold). A3 does not perform like Alternative 1 for C3, therefore the latter will be ranked highest and chosen to be applied to the collection.

The goal is to ensure that during execution of the plan on the entire content set the preservation action continues to perform similar than during experimentation, and expectations and conditions defined in the plan are met.

5.2.3 Statements on Quality

Requirements on performance and quality of outcome have been formalized when specifying utility functions for decision criteria and related measures. This information can be used to derive specifications to check if a measurement meets the expectations. For ordinal measures values and their corresponding target values can be enumerated. For numeric values the key is to use the measurements together with the thresholds.

Taking a look at the evaluation values of A1, following observations lead to statements about expected quality:

1. C1 rejects alternatives for worst value (NO), therefore it must not occur during operations as well: *C1 **must not** be NO.*

2. C2 evaluated for A1 to YES, value NO would have been acceptable, but might have led to a different ranking, and therefore requires re-evaluation if violated: *C2 **should** be YES.*

3. C3 rejects alternatives for measurements < 3 , we could state: *C3 **must not** be less than 3 .*

4. A1 outperforms A3 with a better result for C3, hence a lower value could also require re-evaluation: *C3 **should** be greater than 10.*

These statements satisfy MR.3, and have to be translated to appropriate machine readable interpretation to enable automated monitoring.

5.3 Monitoring of Quality Statements

Where should actual measurements be checked for conformance with these statements of quality?

Decision criteria relate to different aspects of components or the quality of the output. As mentioned earlier, aspects related to the action can be monitored as risks and opportunities using *Scout* directly, including aspects pertaining to the format such as the ISO standardisation of file formats. And measurements related to the outcome of a migration action have to be recorded during migration, and characterisation and quality assurance components have to be applied to determine related measurements.

It seems clear that these should be evaluated immediately. But when considering scalable operations, a distinction between mandatory and desirable levels of quality is necessary:

5.3.1 Desirable Quality Levels

Desirable quality levels can be used to monitor that operations perform within an expected range. Outliers are acceptable, but might require further actions, therefore responsible persons should be notified. Moreover, acceptable changes could have led to a different decision, so re-assessment of the chosen preservation action might be necessary. Examination of these results can be postponed, and can be handled by *Scout*, which already aggregates information from the repository, and Section 5.5 discusses how triggers can be defined.

5.3.2 Mandatory Quality Levels

During operations non acceptable measurements have a different meaning: While the adoption of related preservation actions also needs to be reconsidered, the concerned objects are seen as invalid and must not be ingested to the repository. Taking into account scalability of operations(MR.5) it is not feasible to relay evaluation to *Scout* - it had to pull results via Report API, and the repository had to postpone ingest until the answer arrives. Rather this evaluation should happen close to the location where the components are executed to prevent additional overhead and minimize impact on performance of migration operations.

5.4 Quality Level Descriptions

The quality statements discussed before contain implicit information on the measurements which is required for evaluation. The goal is to provide the evaluation component a set of rules which can be used to validate measurements without the need to refer to the quality ontology for information on types, possible value ranges, and applicable operators. *Plato* already has the knowledge about measures, related scales and their restriction, and specified utility functions, and can use this information to generate enactable descriptions of quality.

In contrary to XML schema languages like RelaxNG and XML Schema(W3C), which provide grammar based validation of XML documents, Schematron[2] is a rule based language that allows to specify assertions on tree patterns in the document. It can enforce the same structure as the before mentioned languages, but is commonly used not as replacement, but to augment schema documents with constraints they cannot impose or to express business rules. The language itself is expressed in XML and consists of only a handful of elements. It builds around human readable assertions, which are augmented with XPath expressions that enforce this verbalized constraints, and are grouped in rules which specify the context where they should hold. A validation against Schematron rules results in a structured output containing the human readable description of the rules that failed, together with information on the responsible elements. There exist native implementations of Schematron, the reference implementation translates Schematron schemas to XSLT stylesheets which can then be used to validate documents.

For these reasons we developed *Plato*'s quality level descriptions based on Schematron schemas.

5.4.1 Evaluation of Quality Level Descriptions

Preservation components as described in Section 4.4 output migration results and measures related to input and output objects. For an efficient evaluation measures should not be validated independently, but in one run. This requires to collect and aggregate measures before validation. The annotations on output ports discussed earlier allow to

[2] `http://www.schematron.com/`

identify ports related to measures. These measures can be combined in one single XML document by the workflow execution platform itself, or the preservation component could be wrapped in another workflow that performs the aggregation. Finally the resulting document can serve as input for a validation component.

An example output is given in Listing 5.1. It shows the list of measures, in this case only related to the output, and their measurement values. For clarification the example names of measures are given in comments.

Listing 5.1: Aggregated output of a preservation component

```
<!DOCTYPE measures [<!ENTITY measures "http://purl.org/DP/quality/measures#">
    ]>
<measures>
   <!-- automated QA supported -->
   <measure subject="outputFile" type="&measures;134">Yes</measure>
   <!-- comparative file size-->
   <measure subject="outputFile" type="&measures;123">2.7</measure>
   <!-- compression type -->
   <measure subject="outputFile" type="&measures;117">none</measure>
   <!-- image width equal -->
   <measure subject="outputFile" type="&measures;51">Yes</measure>
   <!-- image height equal -->
   <measure subject="outputFile" type="&measures;53">Yes</measure>
   <!-- Exif: all tiff data retained -->
   <measure subject="outputFile" type="&measures;251">Yes</measure>
   <!-- colour model preserved -->
   <measure subject="outputFile" type="&measures;56">Yes</measure>
</measures>
```

5.4.2 Generation of Quality Level Descriptions

When generating quality level descriptions first the name of the plan is documented as title of the schema. Then the objective tree is traversed and one rule is added for each criteria that is linked to a measure and has a utility function that can potentially lead to rejection of a preservation action. Its context is defined using the measure's URI and restricting it to output files.

To generate assertions for ordinal values *Plato* iterates over all criteria with ordinal scale, and tests for each potential value if it is unacceptable (that is: has a target value of 0). In that case the XPath location for this rule is extended to avoid this, and a textual description is generated to document the intended reasoning.

Assertions for numerical values are created by iterating over all criteria with numerical scale, and determining if the thresholds of the utility function imply an increasing or decreasing order. According to this, values smaller or equal, or greater or equal threshold 1 can be enforced. Again, a textual description that captures the reasoning is added.

Listing 5.2 shows a reduced example with assertions for measures of boolean, numeric, and ordinal scale. Note that the current implemenation accepts missing values for measures under examination, to prevent the rejection of migration results due to missing quality assurance components.

When validation of these quality level descriptions fails, the whole outcome of a preservation component is marked as failed to prevent ingest of objects which do not meet the quality levels defined during planning. Applied to the aggregated output from above (Listing 5.1), assertion 'comparative file size must be less than or equal to 2.0' would fail. A corresponding error would be reported, and the migration for this object would be marked as failed.

Listing 5.2: Quality level descriptions to enforce levels of quality

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE schema [
   <!ENTITY measures "http://purl.org/DP/quality/measures#">
]>
<schema xmlns="http://purl.oclc.org/dsdl/schematron">
  <pattern>
    <title>QLDs based on plan iPRES 2013 DEMO(test_plan_123)</title>
    <rule context="measure[@type='&measures;134' and (@subject != 'inputFile')
        ]">
      <assert test="(. != 'No')">automated QA supported must have (one) of the
           the following values: [Yes]</assert>
    </rule>
    <rule context="measure[@type='&measures;123' and (@subject != 'inputFile')
        ]">
      <assert test=" . &lt;= 2.0">comparative file size must be less than or
         equal to 2.0</assert>
    </rule>
    <rule context="measure[@type='&measures;117' and (@subject != 'inputFile')
        ]">
      <assert test="(. != 'lossless') and (. != 'lossy')">compression type
         must have (one) of the the following values: [none]</assert>
    </rule>
  </pattern>
</schema>
```

5.5 Triggers

Section 5.2 explained how criteria and transformers embody information on the expected quality of tools, operations, and formats, and how they can be used to derive statements about this quality.

As mentioned earlier, the preservation watch system *Scout* collects information about the world and what is going on within the repository through the means of source adaptors, which aggregate and translate the information to its internal data model. Moreover, *Scout* can process control policies and allows to define triggers to check periodically if a condition is met. As its data model is based on RDF, triggers can be defined by the means of SPARQL queries. The web interface of *Scout* provides some limited support for creation of triggers via templates. But while using these already requires a detailed insight about the internal data model, creating new triggers can be challenging. Thus, we needed to reconstruct the missing documentation ourselves.

As a first step the data model of *Scout* and what information is currently available is analysed, then it is discussed how triggers can be generated automatically to enforce the statements of quality.

5.5.1 Data Model of Scout

Figure 5.1 shows an example of how *Scout* models information about the world and associates it with data collected by its sources. An Entity Type allows to define the information that can be collected about some aspect of the world, and related properties specify the characteristics that can be recorded. Data created by a source adaptor is represented by an entity, which is a concrete instance of some EntityType, and can be related only to property values that have a corresponding property of this type. In this example 'JPEG 2000' is a concrete instance of entity type 'File Format' and has property values for property 'PRONOM Id' and 'Tool support'.

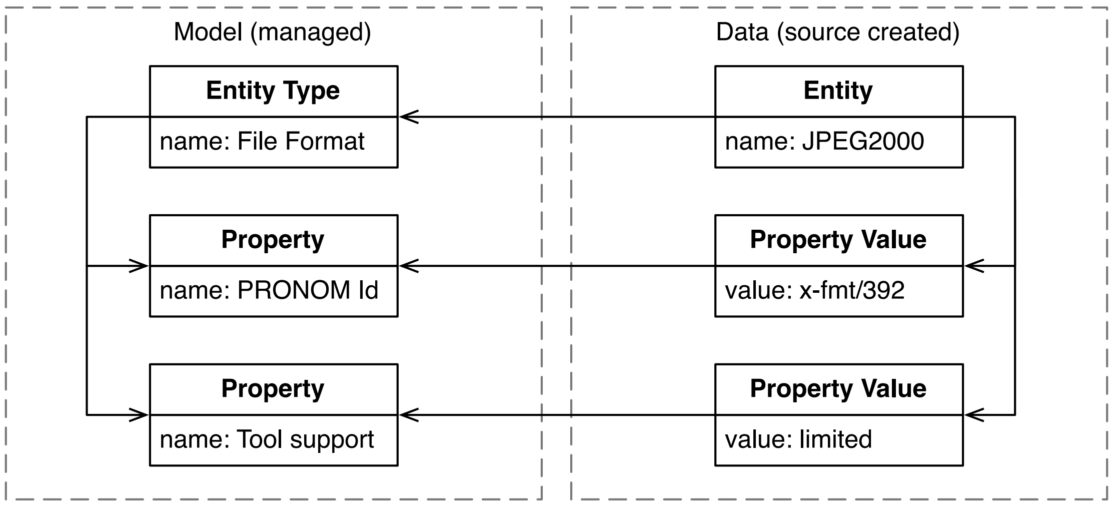


Figure 5.1: *Scout* relating collected data to its internal model[FPD+12]

The example is based on a simplified data model and shows only the most relevant attributes. Each property value is the atomic measurement of a property for an entity measured in a specific point in time. This moment in time is defined by a related measurement that also keeps track of the used data source, and thus allows to capture how certain values change over time, and can reveal trends. Moreover, it is an abstraction and omits some implementation details, thus a closer look at the source code is necessary to specify triggers.

Figure 5.2 gives an overview on *Scout*'s data model that will serve to define triggers. A property has a name and description, and refers to the type of entities it describes, and specifies the data type of its values. Additionally, it can give some renderingHint to the user interface.

Listing 5.3: SPARQL query to retrieve all entities of an entity type

```
PREFIX watch: <http://watch.scape-project.eu/kb#>
PREFIX rdf: <http://www.w3.org/1999/02/22-rdf-syntax-ns#>
SELECT ?s
WHERE {
  ?s rdf:type watch:Entity .
  ?s watch:type ?entityType .
  ?entityType rdf:type watch:EntityType .
  ?entityType watch:name ?name
  FILTER (<entity type> = ?name)
}
```

A property value belongs to an entity, and relates to the property through which it is described. It has one attribute for each supported type, and depending on the data type specified in property only the one holds the actual value.

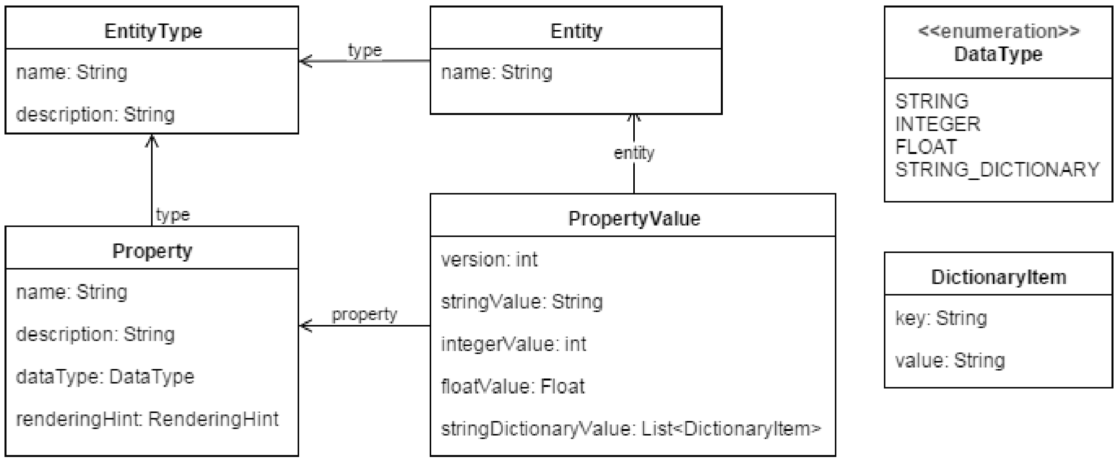


Figure 5.2: Data model of *Scout*

The web interface of *Scout* allows to create advanced queries to explore its content, and to create triggers from these queries. However, the select cause is predefined and limits the result to a list of one of the five data types (EntityType, Property, Entity, PropertyValue, Measurement). Hence the possiblities for both, queries and triggers, are restricted and naturally this limitation will apply as well to generated triggers.

Using the advanced query feature to search for available entity types reveals that the currently stored types are 'content_profile', 'Repository', and 'format'. For each of these types the list of related entities can be retrieved with the query given in Listing 5.3. (Note that the prefixes are only declared in this example for completeness and will be omitted in later listings).

As a next step the information *Scout* collects for each entity is analysed regarding its usefulness for automated monitoring of conditions defined during planning, and we can derive a mechanism to create triggers automatically.

Entity: Content Profile

Content profiles of collections are read via the *c3po* source adaptor, which aggregates the data and translates it to the internal model. Subsequent reads lead to a history of measurements that can give insight into development of formats, and *Scout* provides visualisations to make trends visible. Moreover, *Scout* already creates triggers to check the conformity of content profiles with control policies. Table 5.3 shows the properties stored for content profile *demo*. It can be seen that properties related to individual objects of the collection are aggregated. For numeric values minimum, average, and maximum values are stored. For ordinal measures the distribution of values is stored as key value pairs.

collection size	1.21 GB
compression scheme distribution	5 key-value pairs
format distribution	15 key-value pairs
objects avg size	8.26 MB
objects min size	512 bytes
objects max size	27.27 MB
objects count	206

Table 5.3: *Scout* - Properties of content profile *demo*

These properties are certainly interesting for control policies, for example there could be a directive 'There must be a plan for collections with more than X objects', or 'There must be a plan for collections bigger than Y GB'. But at the moment the quality ontology does not cover aspects related to collections as a whole.

During planning aspects like compression scheme, specific format versions, and also the object size can be of interest, and related measures would exist. But currently these properties do not refer to measures, nor have corresponding names, thus lacking the semantics to generate triggers automatically.

Entity: Format

Scout retrieves and aggregates information about formats from *PRONOM*'s SPARQL endpoint. Available properties can vary depending to the format, Table 5.4 shows the properties related to Acrobat PDF 1.6.

For planning actually interesting information like licence restrictions, licensing costs, or format adoption is missing. Hence data related to this entity is not useful for monitoring conditions.

name	Acrobat PDF 1.6
version	1.6
mime	application/pdf
puid	fmt/20
ext	pdf
formatType	http://reference.data.gov.uk/technical-registry/formatType/Page_Description
byteOrder	http://reference.data.gov.uk/technical-registry/Big_endian
released	Thursday, 1 January 2004
internalSignature	http://reference.data.gov.uk/technical-registry/internalSignature/21
uti	com.adobe.pdf
developedBy	http://dbpedia.org/resource/Adobe_Systems

Table 5.4: *Scout* - Properties of format *Acrobat PDF 1.6*

Entity: Repository

Information on the repository is collected by the Report API adaptor. At the moment this includes events about ingest and migration, and as can be seen in Table 5.5 is very limited. Regarding migration, the adaptor uses all information delivered via the Report API, it can be assumed that once more complete migration actions with integrated quality assurance are in place, the output will be richer, including for example measures related to used compression scheme or similarity metrics.

The large amount of objects and as a result occurring events makes it infeasible to retain the actual values, but require to aggregate them. Property values are stored similarly like for entity content profile: For the numeric property preservation action execution time minimum, average, and maximum values are kept, yet for ingest only the average time. At the moment no ordinal values are stored, but it can be assumed that the distribution of values is also stored as key value pairs, like for entity content profile.

Minimum preservation action execution time	2.056965
Average preservation action execution time	3.8898122
Maximum preservation action execution time	5.984251
Ingest average time (ms)	1092798.0

Table 5.5: *Scout* - Properties of repository *RODA*

5.5.2 Defining Triggers

In the following the creation of triggers based on quality defined during planning is discussed. Here again numeric and ordinal values have to be treated separately.

A proof of concept implementation was integrated in *Plato*. As *Scout* lacks references to measures and uses a naming of properties that does not conform to the quality catalogue, this is currently limited to measure *elapsed time per object*[3].

Triggers Numeric Values

As we have seen numeric values are aggregated, and stored in three separate properties: minimum/average/maximum.

Given a control policy statement, or a decision criteria linked to a measure and with defined transformers, both the aggregated value and operator for the trigger can be determined as shown in Table 5.6. Note that the goal is to detect violations of the policy, hence the operator is negated.

control policy statement	**aggregated value**	**operator**
Value should/must be greater than 100	Minimum	(?fv $\leq$ 100.0)
Value should/must be smaller than 2.0	Maximum	(?fv $\geq$ 2.0)

Table 5.6: Control policies and resulting conditions related to numeric values

Based on this a trigger can be defined that checks for values exceeding the expected ranges. The trigger shown in Listing 5.4 will raise an event when the execution time of a preservation action is greater or equal to two seconds.

Listing 5.4: Trigger for numeric values

```
SELECT ?s
WHERE {
  ?s rdf:type watch:PropertyValue .
  ?entity watch:type ?entityType .
  ?entityType rdf:type watch:EntityType .
  ?entityType watch:name ?entityname.
  ?s watch:property ?property .
  ?s watch:floatValue ?floatValue .
  ?property watch:name ?propertyname .
  FILTER ((?entityname = 'Repository') &&
          (?propertyname = 'Maximum preservation action execution time') &&
           (?floatValue >= 2.0))
}
```

Triggers for Ordinal Values

As explained before at the moment there are no ordinal measurements fed back via the Report API. But the Report API adapter creates a histogram for non-numeric values, stored as key-value pairs, similar the *compression_scheme* distribution is treated for content profiles, e.g.:

```
image orientation preserved { yes : 34, no : 5 }
compression type {none: 12300, lossless : 20, lossy : 0}
```

[3] http://purl.org/DP/quality/measures#11

Table 5.7 shows control policy statements related to ordinal values as *compression type*, and how they could be translated to SPARQL queries:

control policy statement	**occurrence**	**condition**
compression type must not be lossy	lossy	> 0
compression type should not be lossy	lossy	$>$ THRESHOLD
compression type must be none	lossy + lossless + ...	$= 0$

Table 5.7: Control policies and resulting conditions related to ordinal values

Note that there is only an explicit translation for the statement *must not be* to prevent a specific value. In the second case the uncertainty of *should not* needs to be quantified, e.i. a threshold for the acceptable number of outliers needs to be defined. And to ensure a property has only one specific value can only be done indirectly by checking the occurrences of all other possible values the property can have.

The resulting trigger for the latter case is shown in Listing 5.5.

Listing 5.5: Trigger for statement *compression type must be none*

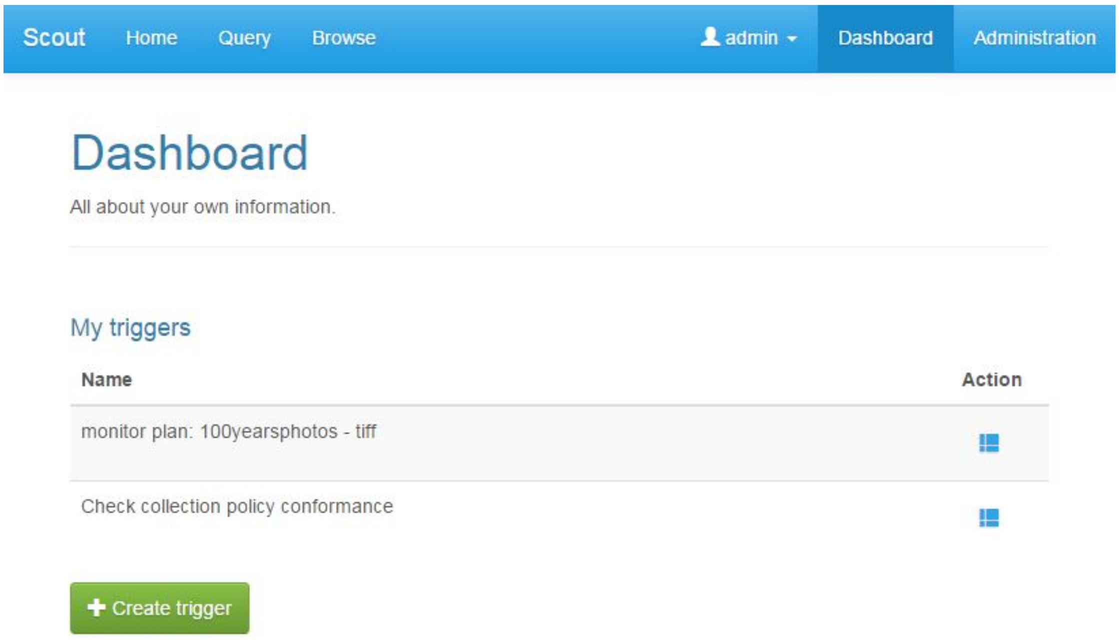
```
SELECT ?s
WHERE {
   ?s rdf:type watch:PropertyValue .
   ?s watch:entity ?collection ;
      watch:property ?compressionSchemeDist ;
      watch:stringDictionaryValue ?value .
   ?collection watch:name "demo"^^xsd:string.
   ?compressionSchemeDist watch:name "compression_scheme distribution"^^xsd:
       string .
   ?value ?l ?dictionaryItem .
   {
   ?dictionaryItem watch:key ?compressionType1;
                 watch:value ?strVal .
     FILTER ((xsd:decimal(?strVal) > 0) &&
           !(?compressionType1 = "None"^^xsd:string))
   }
}
```

Once a plan has been approved, it can be deployed to the repository. Additionally triggers can be automatically generated and deployed to the connected *Scout* instance, as shown in Figure 4.2.

In Figure 5.3 the newly generated trigger can be seen after deployment along with all other active triggers in *Scout*'s trigger overview list, where they can be managed.

5.6 Summary

Continuous monitoring of operations and the environment is necessary to make sure that preservation solutions in place are still performing as expected and are appropriate to the current needs of the user community and technological standards.


Scout Home Query Browse admin Dashboard Administration

Dashboard

All about your own information.

My triggers

Name	Action
monitor plan: 100yearsphotos - tiff	
Check collection policy conformance	

Figure 5.3: Screenshot of *Scout* - Overview of deployed triggers

This chapter first pointed out the requirements of such a monitoring system, before it discussed the quality requirements established during planning and their use to define statements of expected quality levels. It turns out that to maintain scalable operations the category of a measure alone is not enough to decide where the related property should be monitored, but related utility functions have to be considered as well.

Then, generation of automatically checkable statements based on measures and utility functions for two systems was discussed: For immediate validation of migration results by the workflow execution platform quality level descriptions were specified. And taking a closer look at *Scout*'s data model allows to generate triggers to monitor the environment and desirable quality levels.

The data model of *Scout* is very flexible, but at the same time it waives semantics when relying only on textual descriptions of its objects. It uses linked data and understands the quality model, but translate it to its own model, where names of properties are human readable, and relationship to measures is lost. But one have to take into account that it is a first prototype implementation to demonstrate the feasibility of a monitoring system, and these limitations could be overcome by adapting the Report API adaptor delivering actual measures, and extending the model of *Scout* to maintain references to other concepts.

The following chapter presents a case study to evaluate the measures taken in the context of a preservation system, before contributions are summarised and results are discussed in Chapter 7.

CHAPTER 6

Evaluation

In this chapter we evaluate the measures taken to integrate preservation planning with its context. First a case study of a real world preservation plan is summarised, which evaluated only a part of the described changes, before an evaluation scenario is outlined to evaluate the entire modifications.

When comparing different scenarios it has to be considered that the effort required to complete a preservation plan depends on the scenario and the expertise of the involved planners: With knowledge about involved formats and available tools for preservation actions and quality assurance the related planning steps can be carried out faster. Likewise, a homogeneous collection is easier to analyse and sample selection can be done manually, whereas a heterogeneous collection of objects with diverse features is more challenging. If for example only objects of a certain format are accepted for one collection, these are normalized before ingest and might differ only in file size, and selecting the biggest, smallest, and average sized file could be a viable and easy to implement strategy. But if intrinsic features like embedded meta data or tables are present, these should be considered as well for sample selection, a difficult task without proper tool support. Once the objects differ even in format, it is likely that preservation strategies cannot cope with all objects, and the collection has to be partitioned to create separate preservation plans.

In [KBA13] the authors report about a controlled case study conducted with the State and University Library Denmark to create a preservation plan for a collection of recorded radio broadcasts, using *Plato 3*. This was done in a workshop setting: A group of stakeholders and staff members from the library with expertise in policies and strategies, formats, and related tools and quality assurance, was guided by two preservation experts regarding the preservation planning workflow and control of the tool. For every step the required time was tracked separately for related activities, like gathering and analysing the information, and entering it into *Plato*.

Figure 6.1 shows the total effort in person hours per workflow step. In a subsequent step the authors repeated the process in a laboratory environment using *Plato 4*, which contains already part of the changes presented in this work, namely the integration of

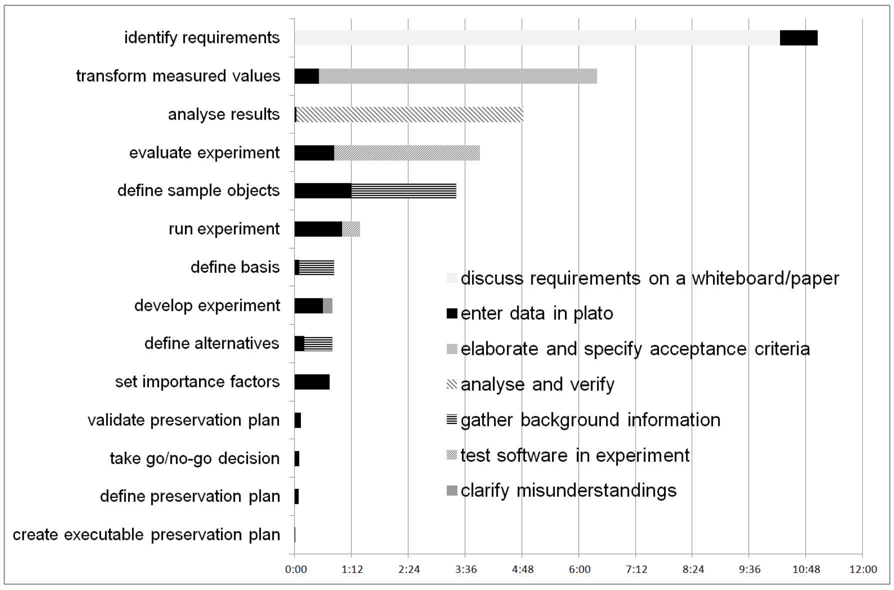


Figure 6.1: Total effort in person hours per workflow step[KBA13]

content profiles from *c3po*, the ability to process control policies and the use of a controlled vocabulary for requirements and evaluation. They conclude that for the presented scenario the overall improvement for entering data into *Plato* is 57%, automation of the content analysis saves 10% of the total planning effort, and effort for discussing requirements and specifying acceptance criteria that makes up for almost half of the total effort could be significantly reduced once control policies are shared across scenarios of the organisation.

Since then the planning tool was developed further and lead to the resulting version *Plato 4.5*. It was part of the SCAPE preservation life-cycle demonstration[DKK$^+$14] that was held at the DL2014 and won the best demo award. In the following a case study on the basis of this demonstration is presented to discuss the effects of the changes in relation to the whole preservation system and to evaluate the usefulness of in term of time saving and automation of processes.

6.1 Evaluation Scenario

Assume an organisation is responsible to keep its content accessible for the public for the next 100 years. Part of this content is a collection of images from national photographers, consisting of files of diverse formats.

Various image formats allow to reduce storage space required for the data by supporting compression algorithms. These can be classified as either lossy or lossless. The former usually provide higher compression rates, at the price of losing information in the process, the original image cannot be restored completely. But also lossless compression poses risks: first the consequences of bit rot are more grave and can affect more and greater areas of the image, second the algorithm to decode the data has to be preserved as well, moreover currently applying licenses have to be considered and future changes thereof could be problematic as in the case of LZW and the GIF file format [1]. Migration to other formats could lead to a loss of image quality due to deficiencies of the used tool or target format, or relevant meta data could get lost in the process, like spatial information or camera settings.

To identify all related risks and determine the best solution to keep the images accessible a preservation plan has to be defined.

This organisation uses the latest version of the SCAPE Planning and Watch suite [KPD+13]: The repository *RODA* is used, because it implements all SCAPE APIs and is connected to an execution environment that supports Taverna workflows for operations. *c3po* aggregates and analyses the characterisation data from the repository, and creates content profiles. Moreover this organisation has defined preservation policies, and translated them to control policies. *Scout* monitors operations, information from file format registries, and gathers information about the content from *c3po* to check their conformance with defined control policies.

Plato 4.5 is used to address this problem and find an optimal preservation solution for the identified files. An instance is set up and a group is configured with the organisations settings: it is connected to the repository, and the RDF representation of the control policies can be loaded.

6.1.1 Define Requirements

In step *Define Basis* information on the organisation and the environment has to be collected. When control policies are present a preservation case can be selected, and information on document types(*Images from national photographers*), organisation(*DEMO organisation*), designated community(*public*), and policies is automatically applied. The objectives of this preservation case cover statements defining the acceptable loss, requirements regarding regarding performance and costs, and minimizing risks. The complete control policy definition serialised as RDF/XML is given in Appendix A.3, examples are:

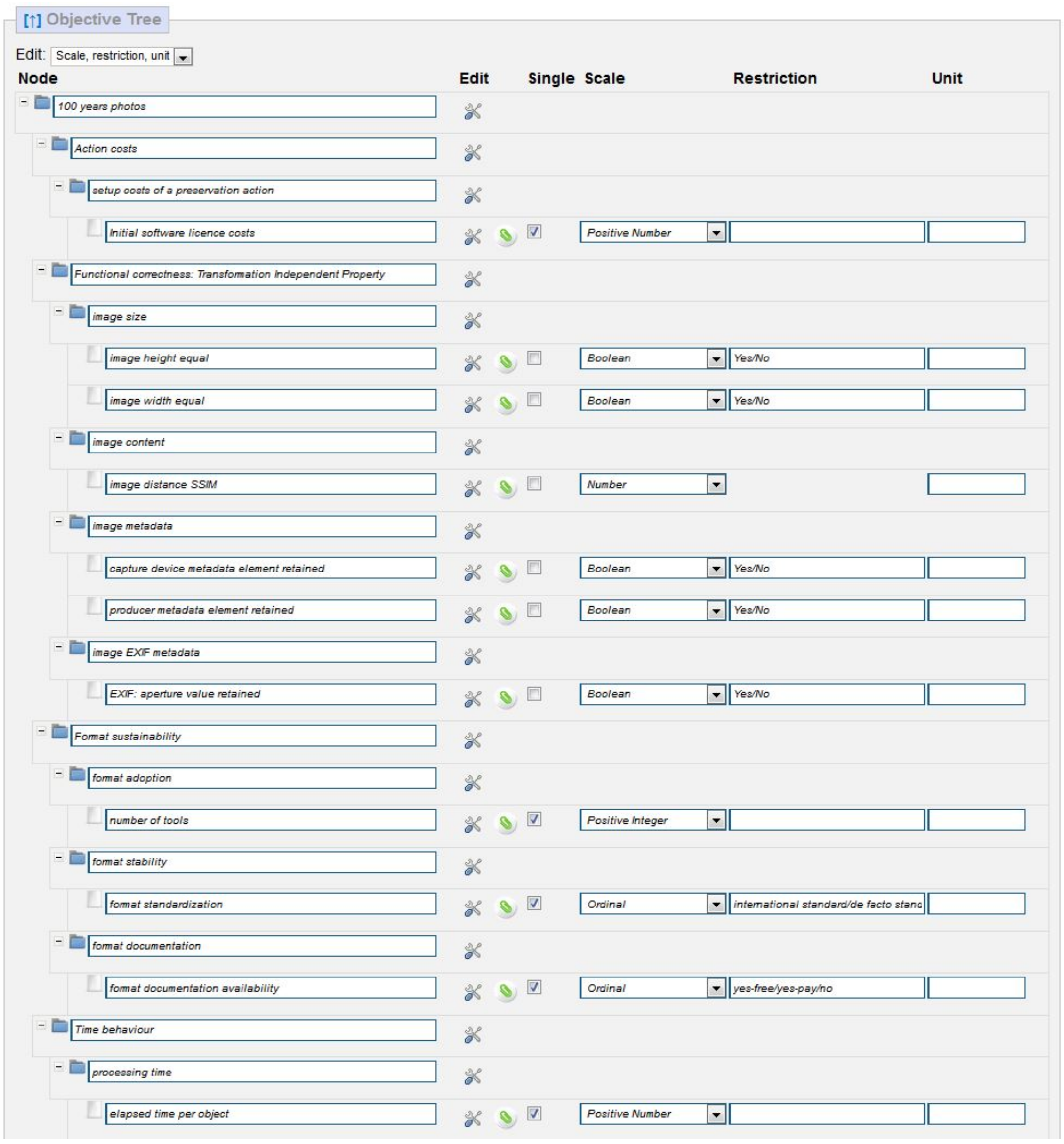
- Acceptable loss includes both, visual properties and information on the objects. To convey the aesthetics the photos should be as close to the originals as possible, as one measure structural similarity index is used: *Image distance SSIM must be greater than 0.98.* Meta data about the conditions under which the photo was taken are of interest for hobby photographers and students and should be preserved: e.g. *EXIF: aperture value retained*

- Budget constraints require to keep the preservation costs low: *Initial software licence costs of an action must be less than 1000 EUR.*

- Considering the volume of the collection processing time is critical: *Elapsed time per object must be less than 10 seconds.*

[1] `https://web.archive.org/web/20090608194513/http://www.unisys.com/about__unisys/lzw/lzw__license__english.htm`

- As mentioned before compression poses a threat to accessibility of the content: *Compression type must be none.*



[↑] Objective Tree

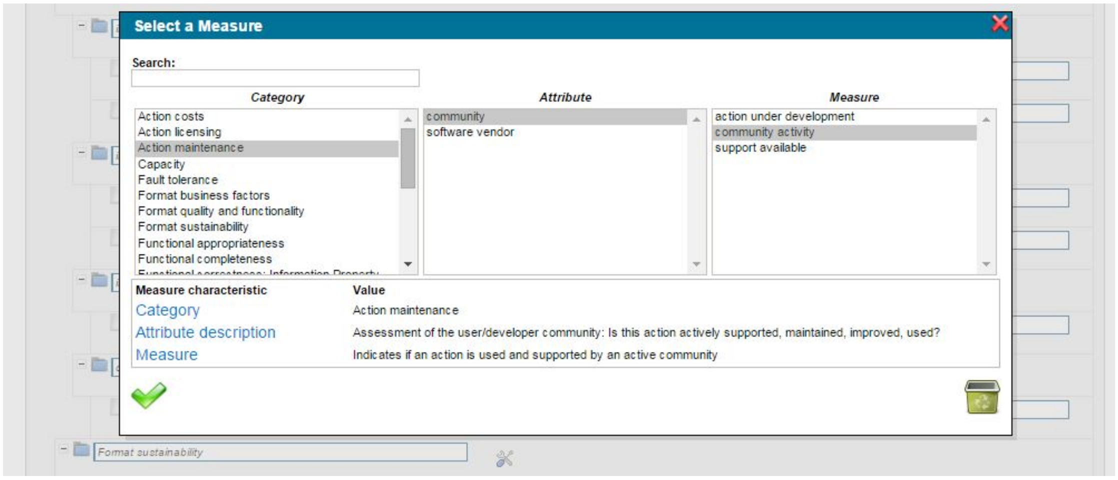
Edit: Scale, restriction, unit

Node	Single	Scale	Restriction	Unit
100 years photos				
Action costs				
setup costs of a preservation action				
Initial software licence costs	✓	Positive Number		
Functional correctness: Transformation Independent Property				
image size				
image height equal	☐	Boolean	Yes/No	
image width equal	☐	Boolean	Yes/No	
image content				
image distance SSIM	☐	Number		
image metadata				
capture device metadata element retained	☐	Boolean	Yes/No	
producer metadata element retained	☐	Boolean	Yes/No	
image EXIF metadata				
EXIF: aperture value retained	☐	Boolean	Yes/No	
Format sustainability				
format adoption				
number of tools	✓	Positive Integer		
format stability				
format standardization	✓	Ordinal	international standard/de facto stand	
format documentation				
format documentation availability	✓	Ordinal	yes-free/yes-pay/no	
Time behaviour				
processing time				
elapsed time per object	✓	Positive Number		

Figure 6.2: Screenshot of *Plato* - Objective tree of case study *100 years photos*

Next, for step *Define Sample Objects* the preservation expert can use the content profiling tool *c3po* to examine the files in question, track down a subset of the collection consisting of jpeg files, and export the collection profile. Two important criteria that have been identified before are preserving meta data and processing time, which is likely related to the file size. Hence, for stratification of the subset a distribution sampling algorithm considering file size and Exif information would be of interest, but currently *c3po* only supports export of content profiles based on its algorithm *Size'o'Matic 3000*, which selects the smallest and the largest objects, and completes the set of samples with objects of size close to the average size. Therefore this is used to create a sample set of size three, and the generated content profile is imported to *Plato*. As a result statistics on the whole collection and the algorithm used to determine representative samples are added to the plan automatically. Moreover, as *Plato* is connected to the repository via the Data Connector API, it uses the REST client to automatically retrieve all samples defined in the collection profile and stores them together with their characterisation information as samples for following experiments.

RODA does not provide the functionality to browse for all meta data of files of a specific format, therefore the internal storage of the repository had to be accessed directly, the whole collection had to be retrieved, or the meta data of the whole collection had to be retrieved and analysed manually. The first option is not in the sense of a trustworthy repository and might be complicated by the way the data is organised internally, and to retrieve the whole collection is clearly not a scalable approach. *c3po* can aggregate the meta data unsupervised, and analysing the collection, generating a collection profile and uploading it to *Plato* can be done in a few minutes.



Action costs
Action licensing
Action maintenance
Capacity
Fault tolerance
Format business factors
Format quality and functionality
Format sustainability
Functional appropriateness
Functional completeness

community
software vendor

action under development
community activity
support available

Measure characteristic	Value
Category	Action maintenance
Attribute description	Assessment of the user/developer community: Is this action actively supported, maintained, improved, used?
Measure	Indicates if an action is used and supported by an active community

Format sustainability

Figure 6.3: Screenshot of *Plato* - Adding criteria during case study *100 years photos*

Arriving at step *Identify Requirements* the planner finds the objective tree already populated as shown in Figure 6.2. Criteria have been automatically synthesised for each objective of the previously selected preservation case, and are organised according to the categories and properties of the referenced measures. These criteria have to be reviewed with respect of their completeness, and if necessary the objective tree could be extended manually. For newly added criteria the catalogue of measures can be browsed and criteria can be created based on suitable measures, which enables automated evaluation and information exchange. Figure 6.3 shows how the criteria catalogue is used to add a measure for community activity, as an active community can support the adoption of a preservation action.

6.1.2 Evaluate Alternatives

Figure 6.4 shows how in step *Define Alternatives* myExperiments can be queried for migration components suitable to handle the selected sample objects. They can be added as alternatives, and provide a full description of the involved tools, their dependencies, applying licenses, and possible parameter settings.

Characterisation and quality assurance components in myExperiment have annotated in- and outputs based on the same vocabulary used for defining decision criteria. Hence, in step *Develop Experiments* suitable components can be looked up and composed automatically (see Figure 4.5), extending migration components to full experimentation workflows. Manual refinement of workflows is supported with the workflow modelling tool Taverna workbench. Two preservation components are composed, based on *ImageMagick 5* and *GraphicsMagick 1.3.16*[2], and available quality assurance components are added.

For step *Run Experiments Plato* allows to automatically execute experiment workflows on a dedicated Taverna server: Sample objects are uploaded together with the workflow to the server, where it is applied to them one by one. Resulting objects together with log output and the measurements from included characterisation and quality assurance components are collected and used to populate the plan. Here again the vocabulary serves to automatically relate outputs of the workflow to criteria in the objective tree and their measurements. Thus in *Evaluate Experiments* only those criteria have to be evaluated manually, for which either no characterisation components exist or that cannot be answered by registries.

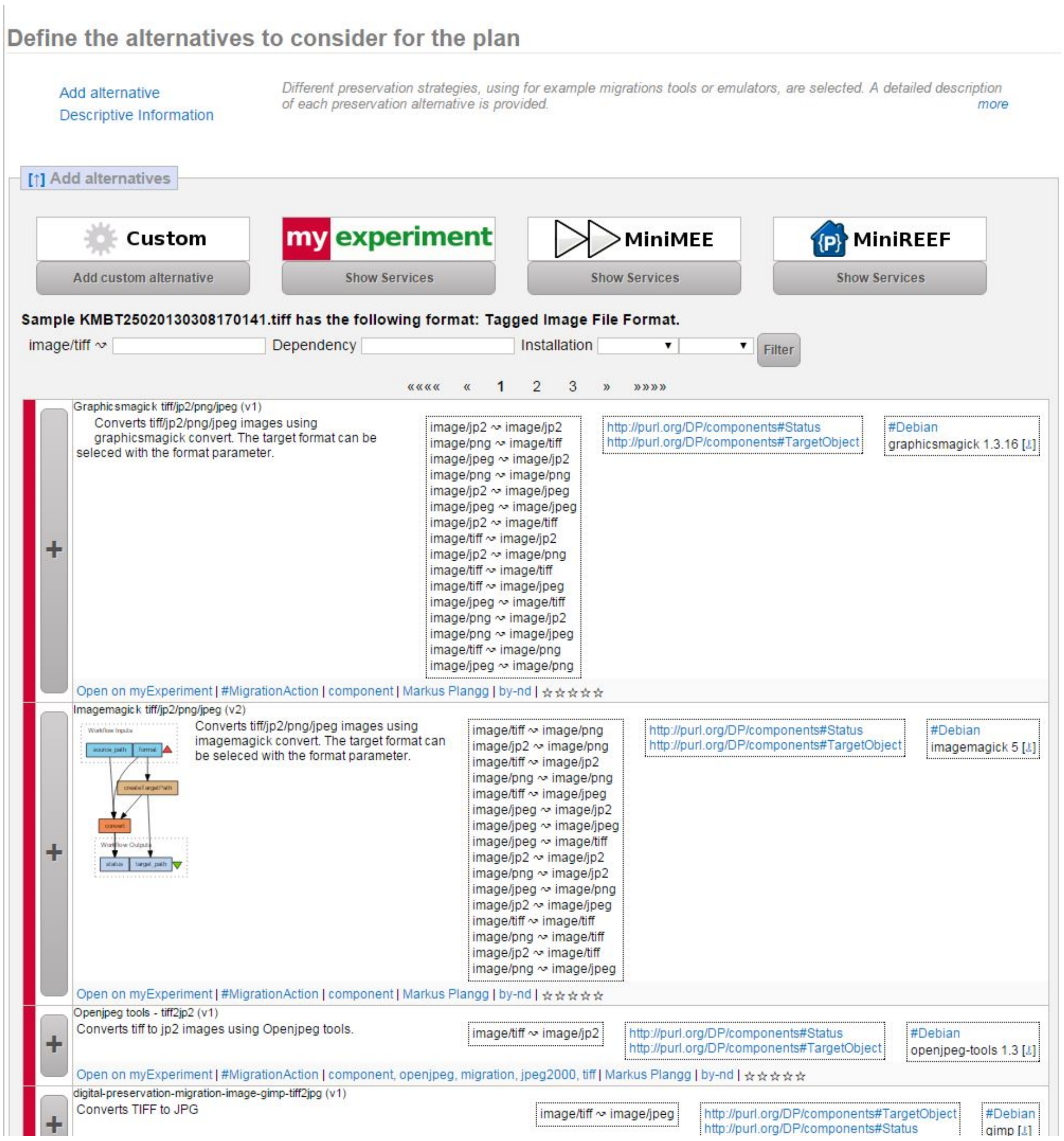
Seven criteria in the objective tree concern the preservation action, and require 14 measurements to be collected. Information on the format, the number of tools available, information on standardization and availability of documentation, and also information on the software licence of the preservation action could be provided by registries. But currently only *number of tools* is covered by the *P2* registry, which reports a wrong result(*no tools available*) due to its static and incomplete knowledge base.

The remaining seven criteria are related to the outcome. Each of it has to be evaluated per action and resulting object. Additionally, for comparative criteria measurements have to be extracted from the sample objects, too. In total 57 measurements need to be collected for criteria related to the outcome.

Four out of this seven criteria are already covered by quality assurance components and amount to 42% of overall required measurements. Their measurements are applied automatically to the plan, and they are available for monitoring of results during operations.

The remaining three criteria are related to image meta data. *Plato* already uses internally *Exiftool* to evaluate image meta data. By wrapping this tool as quality

[2]http://www.graphicsmagick.org/



Define the alternatives to consider for the plan

Add alternative
Descriptive Information

Different preservation strategies, using for example migrations tools or emulators, are selected. A detailed description of each preservation alternative is provided. *more*

[↑] Add alternatives

Custom — Add custom alternative
myexperiment — Show Services
MiniMEE — Show Services
MiniREEF — Show Services

Sample KMBT25020130308170141.tiff has the following format: Tagged Image File Format.

image/tiff ↝ [ ] Dependency [ ] Installation [▾] [▾] Filter

«««« « 1 2 3 » »»»»

Graphicsmagick tiff/jp2/png/jpeg (v1)
Converts tiff/jp2/png/jpeg images using graphicsmagick convert. The target format can be seleced with the format parameter.

image/jp2 ↝ image/jp2
image/png ↝ image/tiff
image/jpeg ↝ image/jp2
image/png ↝ image/png
image/jp2 ↝ image/jpeg
image/jpeg ↝ image/jpeg
image/jp2 ↝ image/tiff
image/tiff ↝ image/jp2
image/jp2 ↝ image/png
image/tiff ↝ image/tiff
image/tiff ↝ image/jpeg
image/jpeg ↝ image/tiff
image/png ↝ image/jp2
image/png ↝ image/jpeg
image/tiff ↝ image/png
image/jpeg ↝ image/png

http://purl.org/DP/components#Status
http://purl.org/DP/components#TargetObject

#Debian
graphicsmagick 1.3.16 [⤓]

Open on myExperiment | #MigrationAction | component | Markus Plangg | by-nd | ☆☆☆☆☆

Imagemagick tiff/jp2/png/jpeg (v2)
Converts tiff/jp2/png/jpeg images using imagemagick convert. The target format can be seleced with the format parameter.

image/tiff ↝ image/png
image/jp2 ↝ image/png
image/tiff ↝ image/jp2
image/png ↝ image/png
image/tiff ↝ image/jpeg
image/jpeg ↝ image/jp2
image/jpeg ↝ image/jpeg
image/jpeg ↝ image/tiff
image/jp2 ↝ image/jp2
image/png ↝ image/jp2
image/jpeg ↝ image/png
image/jp2 ↝ image/jpeg
image/tiff ↝ image/tiff
image/png ↝ image/tiff
image/jp2 ↝ image/tiff
image/png ↝ image/jpeg

http://purl.org/DP/components#Status
http://purl.org/DP/components#TargetObject

#Debian
imagemagick 5 [⤓]

Open on myExperiment | #MigrationAction | component | Markus Plangg | by-nd | ☆☆☆☆☆

Openjpeg tools - tiff2jp2 (v1)
Converts tiff to jp2 images using Openjpeg tools.

image/tiff ↝ image/jp2

http://purl.org/DP/components#Status
http://purl.org/DP/components#TargetObject

#Debian
openjpeg-tools 1.3 [⤓]

Open on myExperiment | #MigrationAction | component, openjpeg, migration, jpeg2000, tiff | Markus Plangg | by-nd | ☆☆☆☆☆

digital-preservation-migration-image-gimp-tiff2jpg (v1)
Converts TIFF to JPG

image/tiff ↝ image/jpeg

http://purl.org/DP/components#TargetObject
http://purl.org/DP/components#Status

#Debian
gimp [⤓]

Figure 6.4: Screenshot of *Plato* - Look up migration actions from myExperiment

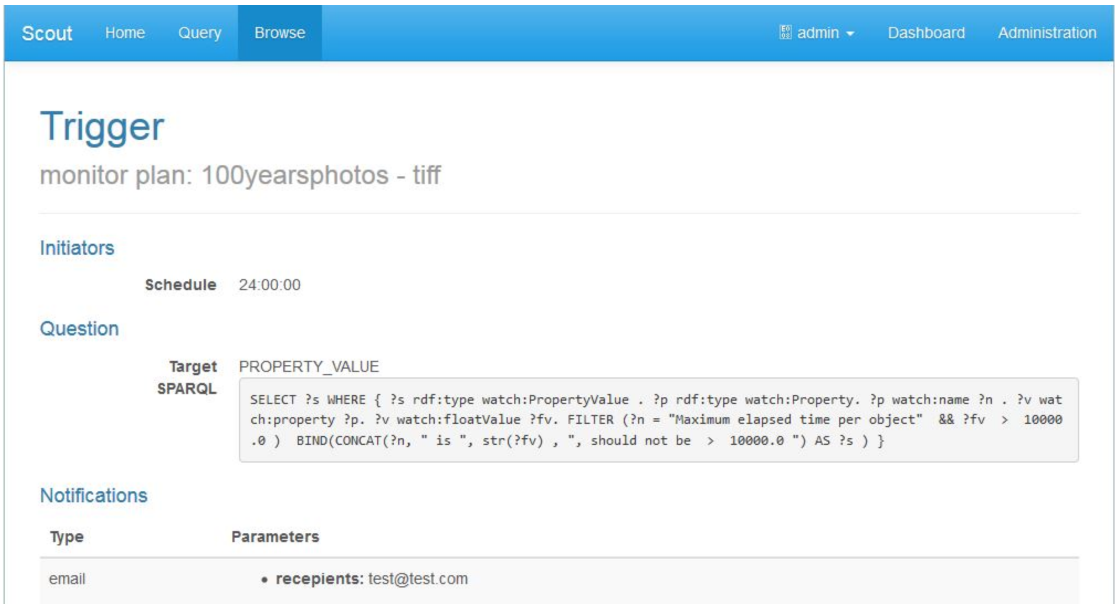
assurance component the coverage of automatically collected measurements could be increased to 80% for the given scenario.

6.1.3 Build Preservation Plan

In *Create Executable Plan* the workflow evaluated in the experiments can be directly used as executable plan, thus ensuring that the very preservation action under test is applied to the content. However, in case part of the characterisation processes are too costly, it might be necessary to limit quality control when running migrations on the whole content. There would exist the possibility to retrieve and edit the workflow, but for the given scenario the quality of results is considered too important to skip quality assurance measures.

Finally a preservation action plan can be generated - assembling information about the subset of the collection from the collection profile, the executable plan, and quality level descriptions which are generated based on criteria and their utility functions - and stored in the plan.

Once the plan is validated and approved, triggers for *Scout* are generated and can be deployed to *Scout* using the implemented interfaces to monitor desirable quality levels (Figure 6.5 shows the trigger monitoring execution time), and the whole preservation plan can be deployed to the repository using the REST client for the Plan Management API. This way the plan itself is preserved in the repository, and can be referenced by preservation meta-data.



Trigger

monitor plan: 100yearsphotos - tiff

Initiators

Schedule 24:00:00

Question

Target PROPERTY_VALUE

SPARQL

```
SELECT ?s WHERE { ?s rdf:type watch:PropertyValue . ?p rdf:type watch:Property. ?p watch:name ?n . ?v wat
ch:property ?p. ?v watch:floatValue ?fv. FILTER (?n = "Maximum elapsed time per object"  && ?fv  >  10000
.0 )  BIND(CONCAT(?n, " is ", str(?fv) , ", should not be  >  10000.0 ") AS ?s ) }
```

Notifications

Type	Parameters
email	• **recepients:** test@test.com

Figure 6.5: Screenshot of *Scout* - Trigger of case study *100 years photos*

6.1.4 Operations and Monitoring

The person responsible for administration of the repository can activate the plan to apply it to the content. This transfers the preservation action plan to the execution environment, where it can be executed without further human intervention. It identifies the targeted objects and contains the workflow which should be applied to them. To achieve the former it includes the information provided by the content profile, a list of

Listing 6.1: Preservation Action Plan - Excerpt of unique object identifiers and quality level descriptions

```
<preservationActionPlan>
  <collection/>
  <objects>
    <object uid="roda:82/roda:83/F32"/>
    <object uid="roda:118/roda:119/F12"/>
    <object uid="roda:118/roda:119/F25"/>
  </objects>
  <executablePlan/>
  <qualityLevelDescription>
   <schema xmlns="http://purl.oclc.org/dsdl/schematron">
    <pattern>
      <title>QLDs based on plan 100yearsphotos - tiff</title>
      <rule context="measure[@type='http://purl.org/DP/quality/measures#1'
          and (@subject != 'SourceObject')]">
        <assert test=" . &gt;= 0.98">image distance SSIM must be greater than
            or equal to 0.98</assert>
      </rule>
    </pattern>
   </schema>
  </qualityLevelDescription>
</preservationActionPlan>
```

unique object identifiers that have to adhere to the specification of the SCAPE Data Connector API. An excerpt of the list is given in Listing 6.1.

It also shows one of the generated quality level descriptions, which will be used during migrations to validate the aggregated output of the preservation component to prevent the ingest of not acceptable results.

Scout monitors the operations of the repository receiving aggregated results of preservation components via the Report API and checking it against defined triggers. Higher resolutions of photos could lead to an increase of file sizes, and consecutive to longer execution times. When they exceed the defined threshold, the generated trigger related to *elapsed time per object* raises an event, and a notification can be stored within *Plato* with the Notification API. The responsible planner has to decide on further actions and revise the plan.

6.2 Observations

The previous sections outlined the creation of a preservation plan when the new features are in place. In the following sections we will discuss the implications and limitations of the changes for the affected steps to evaluate the planning tool objectively.

6.2.1 Define Basis

At the moment control policies do not cover more information than document types, organisation, designated community, and policies, but clearly they could be extended. It can be argued that the control policy definition has to be done specifically for one preservation case, and therefore actual re-use of information and time saving is not given. But high level information about the mandate, relevant organisational procedures and workflows, and contracts should be already available in the organisation, and could be compiled before, and creation of policy statements could be automated. This separation could also reflect that there are usually different roles involved in specifying strategies and defining preservation plans.

While the discussed changes make room for more improvements, they currently do not save time for the step itself, but prepare the plan for subsequent steps.

6.2.2 Define Sample Objects

When *c3po* is used for analysing the content and sample selection, and *Plato* is connected to a SCAPE repository, this step is completely automated. As mentioned before the related effort with these tasks and thus also the actual time saving depend on the complexity of the collection in question. Even for a simple case like in the case study described earlier it can account for a tenth of the overall effort. Analysing a more diverse collection requires to consider a broader spectrum of features to determine a representative set, hence bears a greater potential for saving time.

Note that the data model of *Plato* currently only supports simple binary objects, at the moment binary data of digital objects composed of multiple files cannot be handled. As a consequence automated migrations are limited to one-to-one migrations, even the case of grouping a set of scanned pages of a book to a single PDF file is not possible. Therefore a plan for such objects does not profit from this automation.

6.2.3 Identify Requirements

As can be seen in Figure 6.1 a main part of the planning process is spent on discussing requirements and defining acceptance criteria. The catalogue of well defined measures can help to specify requirements and fosters reuse within and across organisations. Based on real word case studies, it currently covers 415 measures related to different aspects and content types, and is published and publicly accessible, but modifications are moderated.

In the presented case study a complete objective tree is derived automatically from control policies, which reduces the time for this step significantly. Clearly, specifying control policies and related objectives is equally effort intensive, and simply relocating this activity does not save time when done for one preservation plan only. But as pointed out in [KBA13] control policies determined by requirements of the organisation or the designated community are likely to apply to multiple scenarios, can be reused easily for other preservation plans, and *Plato* allows to share control policy between users.

6.2.4 Evaluate Alternatives

When using preservation components the whole task of evaluating alternatives - identifying and describing migration actions, configuration of experiments and execution thereof - is automated and human interaction is reduced to selecting components from a list and providing parameter settings. The execution of experiment workflows can be started by hitting a play button, and criteria related to the outcome of actions can be evaluated automatically.

Manual evaluation would require to extract image width and height from the three sample objects and six migration results and compare the characterisation results manually, run tools to calculate compression type and the SSIM index on all six migration results, and finally enter the data in *Plato*.

Naturally, the involved decisions - which preservation actions to consider, and if the evaluation should be carried out - have to be left to the user and cannot be automated.

These improvements are limited to migration actions, and depend on the availability of migration, characterisation, and quality assurance components in myExperiment. Mapping tools as components and providing the necessary annotations requires additional effort, but this could be compensated by reusing workflows within the organisation, or sharing them on myExperiment with the preservation community, which has a positive attitude towards knowledge sharing. Furthermore, the system providing the Taverna server for running experiments needs to be set up with all tools invoked by the workflows. This could be automatised for tools with package manager configurations, but currently requires manual preparation.

Preservation actions like emulation, keep the status quo, or actions related to complex objects have to be described and evaluated manually.

6.2.5 Build Preservation Plan

To assess the improvements of these steps a look at the case study of the report does not help, as the decision in this plan was to take no action but monitor the environment for changes that would require a re-evaluation. Hence, no time had to be spent on creating an executable plan, which is visible in Figure 6.1.

Before the presented changes the executable plan had to describe the preservation action detailed enough to make a later implementation in the repository possible. Both steps had to be done manually, and the preservation plan only informed the person responsible for operations, without guarantee that the preservation component was adopted as tested. Moreover, monitoring of results was either not available, or conditions had to be defined ad hoc, unrelated to quality definitions specified during planning.

Now a preservation action plan can be automatically created from the preservation component under test, including quality assurance components, monitoring conditions that are derived automatically from the specified requirements, and information about the related objects that is taken from the content profile. It can be deployed directly to the repository, where it is ready to be activated. Additionally, triggers can be generated to monitor the environment, and deployed directly to *Scout*.

6.2.6 Operations and Monitoring

Part of the preservation plan is the preservation action plan, which can be extracted and passed on as concise representation of all necessary information to apply the preservation action to the content.

Information about the target objects stems from the content profile, which at the moment provides a list of identifiers. This approach works as proof of concept, but does not scale for larger content sets. The Data Connector API, which is used by the execution platform to retrieve the objects, specifies interfaces that accept SRU queries, but currently these are neither supported by *RODA* nor by *fedora 4*. The usage of SRU queries is already foreseen in the schema of the preservation action plan.

A methodology was presented to derive monitoring conditions from requirements and related utility functions. Quality level descriptions are created for criteria that are related to the outcome of migrations and with utility functions that could lead to the rejection of an alternative. Based on the presented vocabulary and encoded as Schematron rules they are implementation independent, but depend on the existence of quality assurance tools that deliver measures. Triggers are created as SPARQL queries and can be deployed directly to *Scout*. This works as proof of concept, but the ad hoc naming of stored properties in *Scout* hinders automatic trigger creation. The Notification API has been specified and implemented to store notifications in *Plato*, and a Re-Assessment API has been specified to evaluate a plan with respect to the change of a measurement. Currently these interfaces are not used by *Scout*, which still notifies planners only via emails. For a re-assessment *Scout* lacks the link of measures to the related plan.

6.2.7 Summary

The previous sections discussed for each affected workflow step the improvements and limitations of presented changes. Not covered were phase *Analyse results*, and steps *Validate preservation plan* and *Define preservation plan*. Even though a methodology to derive utility functions has been outlined it was not implemented as the results were expected to be too fragmentary, and the remaining steps are related to cost calculations or depend on the judgment of the planner. Two effort intensive steps could be completely automated: Collection analysis and sample selection is supported by integration with content profiling tools and the repository (G.2/G.4), and decision criteria are derived from control policies (G.1). Adopting a controlled vocabulary to describe aspects of preservation quality fosters reuse through control policies and supports automatic evaluation. It provides further advantages for migration strategies as it enables lookup and automatic composition of preservation components (G.3): This speeds up phase *Evaluate alternatives* significantly when migration and quality assurance components are available, and allows to deploy the selected strategy as part of the Preservation Action Plan directly to the repository, using the Plan Management API (G.4). Monitoring conditions are generated to enforce quality of operation results, and to monitor the environment (G.3), the latter currently as proof of concept.

CHAPTER 7

Conclusions

This chapter summarises first the contributions of this work, before finally results and future work are discussed.

7.1 Contributions

Organisational Policies

Various organisations have the mandate to preserve electronic documents and keep them accessible in the future for a designated community. They formulate their goals and constraints they have to cope with as policies, usually high level guidelines written down in natural language. While useful as basis for a structured approach and for documentation purposes, the embodied information is not accessible to supporting tools.

In this work an ontology was identified which can be used to formulate machine enactable policies. The planning tool *Plato* was adapted to read and process these so called control policies. They can be stored for groups to ease collaboration, and are then available during the planning process. Contained information is used for documentation, like the designated community and organisation. Moreover, its requirements and constraints are used to synthesize a complete objective tree with its decision criteria, the foundation for the evaluation of preservation actions.

Decision Criteria

The definition of measurable decision criteria is the basis for an objective evaluation of alternatives and as such the component selection problem. Starting with user definable criteria collected during case studies and using early versions of *Plato* provided insight and resulted in a valuable body of knowledge. But soon it was clear that this approach is problematic, as it hinders knowledge sharing, prevents automatic evaluation and as a result monitoring of results, and complicates to provide guidance in the process of requirements definition. Efforts were made to structure and classify the various types of measurements, and the resulting criteria catalogue helped to analyse the problem, but was rigid and not easy accessible. This criteria catalogue was reorganised based on a the standard Systems and software Quality Requirements and Evaluation, and in a further step remodelled using RDF, which facilitates linking elements with other models.

Part of this work was to adjust the data model of *Plato* in order to use this vocabulary to represent properties of quality for decision criteria. This provided the basis for further improvements: The integration of control policies, generation of quality level descriptions to enable automated quality assurance of results of operations, and triggers for *Scout* for continuous monitoring of results. Moreover, it allowed others to improve automated lookup of characterisation components, and automated composition of preservation components and thus to improve automated evaluation.

Content Profiling

A profound knowledge of the content and its peculiarities is necessary to decide how to preserve it. As collections are usually too big to conduct tests on all objects, representative samples have to be selected. This set should be as small as possible, but at the same time cover characteristics of the whole collection. To determine such a set is not a trivial task, and requires tool support.

This work demonstrates how content profiling tools can be integrated in the preservation planning process. As a proof *Plato* was extended to read content profiles generated by *c3po*. Contained information about the collection and representative samples are automatically added to the preservation plan as evidence. Moreover, if *Plato* is connected to a repository, these samples are automatically retrieved and available for later experiments.

Interoperability

The importance of preservation planning for organisations with the goal to preserve their content for the long term is reflected in its role in the OAIS standard. Yet this standard does not specify any implementation details on components nor interfaces. So far no standards for interfaces to interact with repositories exist. The project SCAPE aimed to develop an environment of loosely coupled components to achieve a scalable preservation environment. One result was a set of open interfaces to increase interoperability between repository and preservation components, and among preservation components themselves.

In this work *Plato* was extended to utilise these interfaces: The Data Connector API is used to retrieve sample objects, and the Plan Management API to deploy preservation plans to repositories. Therefore it can be integrated with all repositories that provide these interfaces. Part of defining these interfaces was to describe the exchanged data. The so called Preservation Action Plan was specified, an XML representation of all information required to apply a preservation action to a content set and monitoring the results. Moreover, the Notification API was specified and implemented, which allows integration with monitoring components like *Scout*.

7.2 Discussion and Future Work

The previous section summarised contributions of this work. Most of these changes touch only part of the preservation planning workflow. But the case study described earlier shows how a common language for key concepts together with better integration with existing systems can benefit not only the planning process but also enable continuous preservation of content. Naturally, the decision to give up control and not try to develop a functional complete system, but instead enable integration with existing system and tools through open interfaces and a common language, presents a weakness: The actual impact of the changes presented in this work depends on the acceptance of this language by the preservation community, and adoption of the APIs by repository developers.

At the moment *Scout* does send notifications only by email, and re-assessment of plans is not considered yet. It understands control policies in the way that it can create triggers to monitor conformance of gathered information, but its internal data model focuses on human readable names and does not maintain references to measures, which hinders automated creation of triggers. The model has to be analysed, missing concepts identified, and adapted. As it is based on linked data, it could be extended to reference additional information like measures.

Objectives of control policies are related to a content set, but this concept is currently not used in the systems discussed in this work. As a consequence the objects affected by a preservation plan have to be listed explicitly, which is not a scalable solution. But preservation action plans are already designed to accept also queries based on the Search/Retrieve via URL (SRU)[1] protocol. These could be provided by *c3po*'s content profile together with identifiers, and *Scout* should refer to these so triggers could be defined specifically for a subset of the content. In turn, the Report API adaptor has to generate information about which collection the measurements belong to. Additionally, information about the related preservation plans should be reported back to *Scout*, to enable re-assessments.

The presented changes provide further opportunities for improvement of the preservation tool itself:

Plato provides the possibility to browse the quality catalogue to use for decision criteria, and it allows to specify user defined criteria. The latter are unknown to other planners, and are not available for automatic evaluation, reuse, and derivation of knowledge. On the other hand the quality catalogue is openly accessible, but only in read only mode. A user-based management of this catalogue, accessible from within *Plato*, could ease access and support growth, yet mechanism to curate these entries would have to be in place, as the definition of meaningful and concise measures can be difficult.

Evaluation of preservation solutions is time consuming and effort intensive work. It is desirable to automate this as far as possible, but tool support and existing registries will and cannot cover all criteria. For example costs to employ a strategy might depend on the organisation itself, and in the case of emulation automating the evaluation is challenging. Nevertheless, the adopted language fosters analysing existing plans, and

[1] `http://www.loc.gov/standards/sru/`

importance factors could be used for more sophisticated ways of presenting the criteria under evaluation, as suggested in [BKPR13].

Another limitation is the lack of support for more complex objects, e.g. for whole web-pages in the case of web archiving. *Plato* can be used to evaluate preservation actions to deal with harvested content, but stripped off the improvements discussed in this work. Concepts have to be developed to represent such digital objects and deal with them. They should be aligned with current attempts to standardise data exchange with repositories, like the SCAPE Digital Object model.

APPENDIX A

Appendices

A.1 XML Schema of Preservation Action Plan

Listing A.1: XML-Schema for Preservation Action Plan

```
<?xml version="1.0" encoding="UTF-8"?>
<!-- Plato: Executable Plan
   Version: 1.1
     * added QLDs
     * added id for executable plan
   Author: Michael Kraxner
 -->
<xs:schema xmlns:xs="http://www.w3.org/2001/XMLSchema" xmlns="http://ifs.tuwien.ac.at/
    dp/plato" xmlns:tav="http://taverna.sf.net/2008/xml/t2flow" targetNamespace="http
    ://ifs.tuwien.ac.at/dp/plato" elementFormDefault="qualified" version="1.1">
  <xs:import namespace="http://taverna.sf.net/2008/xml/t2flow" schemaLocation="t2flow.
      xsd"/>
  <xs:element name="preservationActionPlan" type="PreservationActionPlan"/>
  <xs:complexType name="PreservationActionPlan">
    <xs:annotation>
      <xs:documentation>
        Describes a preservation action plan:
        * Identifies the collection it was created for,
        * the objects it should be applied to,
        * the executable plan itself, and
        * the expected quality criteria of the resulting object
        TBD:
        * More information on the preservation plan?
        * describe when the plan has to be applied (on ingest, to existing objects...)
      </xs:documentation>
    </xs:annotation>
    <xs:sequence>
      <xs:element name="collection" type="Collection"/>
      <xs:element name="objects" type="Objects"/>
      <xs:element name="executablePlan" type="ExecutablePlan"/>
      <xs:element name="qualityLevelDescription" type="QualityLevelDescription"
          minOccurs="0"/>
    </xs:sequence>
    <xs:attribute name="schemaVersion" type="Version1" />
```

```
</xs:complexType>
<xs:complexType name="Collection">
  <xs:annotation>
    <xs:documentation> Identifies the collection this action plan should be applied to.
           </xs:documentation>
  </xs:annotation>
  <xs:sequence/>
  <xs:attribute name="uid" type="xs:string" use="required"/>
  <xs:attribute name="name" type="xs:string"/>
</xs:complexType>
<xs:complexType name="Objects">
  <xs:annotation>
    <xs:documentation> Defines the objects this action plan should be applied to.
    This can be done either explicit, by providing a list of objects, or by providing
         an SRU query. </xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:sequence>
      <xs:element name="object" type="Object" maxOccurs="unbounded"/>
    </xs:sequence>
    <!--
  <xs:element name="query" type="ObjectQuery"/>
      -->
  </xs:choice>
</xs:complexType>
<xs:complexType name="Object">
  <xs:annotation>
    <xs:documentation>The object, identified by a unique identifier which must be
         suitable for the SCAPE Data Connector API</xs:documentation>
  </xs:annotation>
  <xs:sequence/>
  <xs:attribute name="uid" type="xs:string" use="required"/>
</xs:complexType>
<xs:complexType name="QualityLevelDescription">
  <xs:annotation>
    <xs:documentation>Quality Level Description for characterization and quality
         assurance outputs of the migrated object.</xs:documentation>
  </xs:annotation>
  <xs:sequence>
    <xs:any namespace="http://purl.oclc.org/dsdl/schematron" processContents="skip"></
         xs:any>
  </xs:sequence>
</xs:complexType>
<xs:complexType name="ExecutablePlan">
  <xs:annotation>
    <xs:documentation> The executable plan itself, which can be applied to objects. </
         xs:documentation>
  </xs:annotation>
  <xs:choice>
    <xs:element ref="tav:workflow">
      <xs:annotation>
        <xs:documentation>
          A taverna workflow adhering to the SCAPE component profile Executable Plan
          see: https://github.com/openplanets/scape-component-profiles/blob/master/
               profiles/MigrationExecutablePlanComponentProfile.xml
      </xs:documentation>
      </xs:annotation>
    </xs:element>
  </xs:choice>
  <xs:attribute name="type" type="ExecutablePlanType"/>
  <xs:attribute name="id" type="xs:string"/>
</xs:complexType>
```

```
  <xs:simpleType name="ExecutablePlanType">
    <xs:annotation>
      <xs:documentation>The type of the executable plan, at the moment only taverna
          workflows are supported</xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:enumeration value="t2flow"/>
    </xs:restriction>
  </xs:simpleType>
  <xs:simpleType name="Version1">
    <xs:annotation>
      <xs:documentation>Version of preservation action plan covered by this schema -
        restricted to "1" or "1.x" - for instance "1.2" would be allowed.
      </xs:documentation>
    </xs:annotation>
    <xs:restriction base="xs:string">
      <xs:pattern value="1(|\..*)"/>
    </xs:restriction>
  </xs:simpleType>
</xs:schema>
```

The current version of this schema is also published on the official website of *Plato* [1].

A.2 Example Content Profile

Listing A.2: *c3po* content profile

```
<?xml version="1.0" encoding="UTF-8"?>
<profile xmlns="http://ifs.tuwien.ac.at/dp/c3po" xmlns:xsi="http://www.w3.org/2001/
    XMLSchema-instance" collection="demo" date="Wed Jul 23 09:05:29 WEST 2014" count="
    206">
  <partition count="104">
    <filter id="61d29510-9031-41be-8f21-6cac7192424f">
      <parameters>
        <parameter>
          <name>collection</name>
          <value>demo</value>
        </parameter>
        <parameter>
          <name>metadata.mimetype.value</name>
          <value>image/tiff</value>
        </parameter>
      </parameters>
    </filter>
    <properties>
      <property id="format" type="STRING" count="104">
        <item id="Tagged Image File Format" value="104"/>
      </property>
      <property id="mimetype" type="STRING" count="104">
        <item id="image/tiff" value="104"/>
      </property>
      <property id="lastmodified" type="DATE" count="104">
        <item id="2014" value="104"/>
      </property>
      <property id="creating_application_name" type="STRING" count="104"/>
      <property id="size" type="INTEGER" count="104" sum="1.64463042E9" min="1.2214643E7
          " max="2.7818375E7" avg="1.5813754038461538E7" var="9.092596943307848E12" sd="
          3015393.3314424916"/>
      <property id="checksum_md5" type="STRING" count="104"/>
```

[1] http://ifs.tuwien.ac.at/dp/plato/schemas/preservationActionPlan-V1.xsd

```xml
 <property id="lastmodified_fs" type="DATE" count="104">
  <item id="2014" value="104"/>
 </property>
 <property id="original" type="STRING" count="104"/>
 <property id="compression_scheme" type="STRING" count="104">
  <item id="LZW" value="102"/>
  <item id="Uncompressed" value="2"/>
 </property>
 <property id="image_width" type="INTEGER" count="104" sum="480576" min="4416" max=
    "4672" avg="4620.923076923077" var="4992.378698224848" sd="70.65676682544176"/
    >
 <property id="image_height" type="INTEGER" count="104" sum="620256" min="5872" max
    ="6016" avg="5964" var="1595.0769230769233" sd="39.9384141282165"/>
 <property id="colorspace" type="STRING" count="104">
  <item id="BlackIsZero" value="104"/>
 </property>
 <property id="orientation" type="STRING" count="104"/>
 <property id="wellformed" type="BOOL" count="2">
  <item value="true" count="2"/>
  <item value="false" count="0"/>
 </property>
 <property id="sampling_frequency_unit" type="STRING" count="104"/>
 <property id="sampling_frequency_x" type="FLOAT" count="104" sum="31200" min="300"
     max="300" avg="300" var="0" sd="0"/>
 <property id="sampling_frequency_y" type="FLOAT" count="104" sum="31200" min="300"
     max="300" avg="300" var="0" sd="0"/>
 <property id="bitspersample" type="STRING" count="104"/>
 <property id="samplesperpixel" type="INTEGER" count="104" sum="104" min="1" max="1
    " avg="1" var="0" sd="0"/>
 <property id="scanner_manufacturer" type="STRING" count="104"/>
 <property id="scanner_modelname" type="STRING" count="104"/>
 <property id="scanner_softwarename" type="STRING" count="104"/>
 <property id="valid" type="BOOL" count="2">
  <item value="true" count="2"/>
  <item value="false" count="0"/>
 </property>
 <property id="byteorder" type="STRING" count="2">
  <item id="Unknown" value="102"/>
  <item id="little endian" value="2"/>
 </property>
 <property id="imageProducer" type="STRING" count="2"/>
</properties>
<samples type="size'o'matic 3000">
 <sample uid="roda:71/R2013-03-22T17.20.40.45Z/F6">
  <record name="puid" value="fmt/7" tool="Droid 3.0"/>
  <record name="puid" value="fmt/8" tool="Droid 3.0"/>
  <record name="puid" value="fmt/9" tool="Droid 3.0"/>
  <record name="puid" value="fmt/10" tool="Droid 3.0"/>
  <record name="format" value="Tagged Image File Format" tool="file utility 5.09"/>
  <record name="mimetype" value="image/tiff" tool="file utility 5.09"/>
  <record name="lastmodified" value="Tue Jul 22 18:32:13 WEST 2014" tool="Exiftool
     9.06"/>
  <record name="creating_application_name" value="RollFilm Ver 2.2n" tool="Exiftool
      9.06"/>
  <record name="size" value="15961383" tool="OIS File Information 0.1"/>
  <record name="checksum_md5" value="bdc7c606541e1e81c7dfe7c86dde343d" tool="OIS
     File Information 0.1"/>
  <record name="lastmodified_fs" value="Tue Jul 22 18:32:13 WEST 2014" tool="OIS
     File Information 0.1"/>
  <record name="original" value="true" tool="RODA 1.0.0"/>
  <record name="compression_scheme" value="LZW" tool="Exiftool 9.06"/>
  <record name="image_width" value="4672" tool="Exiftool 9.06"/>
```

```
      <record name="image_height" value="5936" tool="Exiftool 9.06"/>
      <record name="colorspace" value="BlackIsZero" tool="Exiftool 9.06"/>
      <record name="orientation" value="normal*" tool="Exiftool 9.06"/>
      <record name="sampling_frequency_unit" value="inches" tool="Exiftool 9.06"/>
      <record name="sampling_frequency_x" value="300.0" tool="Exiftool 9.06"/>
      <record name="sampling_frequency_y" value="300.0" tool="Exiftool 9.06"/>
      <record name="bitspersample" value="8" tool="Exiftool 9.06"/>
      <record name="samplesperpixel" value="1" tool="Exiftool 9.06"/>
      <record name="scanner_manufacturer" value="Wicks and Wilson" tool="Exiftool 9.06"
          />
      <record name="scanner_modelname" value="RS300" tool="Exiftool 9.06"/>
      <record name="scanner_softwarename" value="RollFilm Ver 2.2n" tool="Exiftool 9.06
          "/>
      <record name="format_version" value="3" tool="Droid 3.0"/>
      <record name="format_version" value="4" tool="Droid 3.0"/>
      <record name="format_version" value="5" tool="Droid 3.0"/>
      <record name="format_version" value="6" tool="Droid 3.0"/>
    </sample>
   </samples>
   <elements>
    <element uid="roda:79/R2013-03-22T17.17.51.70Z/F7"/>
    <element uid="roda:79/R2013-03-22T17.17.51.70Z/F11"/>
    <element uid="roda:71/R2013-03-22T17.20.40.45Z/F6"/>
   </elements>
  </partition>
</profile>
```

A.3 Example control policies

Listing A.3: Control policies for case study *100-Years-Photos collection*

```
<?xml version="1.0" encoding="UTF-8"?>
<!DOCTYPE rdf:RDF [
   <!ENTITY quality "http://purl.org/DP/quality#">
   <!ENTITY measures "http://purl.org/DP/quality/measures#">
   <!ENTITY preservation-case "http://purl.org/DP/preservation-case#">
   <!ENTITY control-policy "http://purl.org/DP/control-policy#">
   <!ENTITY modalities "http://purl.org/DP/control-policy/modalities#" >
   <!ENTITY qualifiers "http://purl.org/DP/control-policy/qualifiers#" >
   <!ENTITY org "http://www.w3.org/ns/org#" >
   <!ENTITY foaf "http://xmlns.com/foaf/0.1/" >
   <!ENTITY owl "http://www.w3.org/2002/07/owl#" >
   <!ENTITY xsd "http://www.w3.org/2001/XMLSchema#" >
   <!ENTITY rdfs "http://www.w3.org/2000/01/rdf-schema#" >
   <!ENTITY rdf "http://www.w3.org/1999/02/22-rdf-syntax-ns#" >
]>
<rdf:RDF xmlns="http://www.demo.org/policies#"
   xml:base="http://www.demo.org/policies/"
   xmlns:quality="http://purl.org/DP/quality#"
   xmlns:measures="http://purl.org/DP/quality/measures#"
   xmlns:preservation-case="http://purl.org/DP/preservation-case#"
   xmlns:control-policy="http://purl.org/DP/control-policy#"
   xmlns:modalities="http://purl.org/DP/control-policy/modalities#"
   xmlns:qualifiers="http://purl.org/DP/control-policy/qualifiers#"
   xmlns:org="http://www.w3.org/ns/org#"
   xmlns:skos="http://www.w3.org/2004/02/skos/core#"
   xmlns:foaf="http://xmlns.com/foaf/0.1/"
   xmlns:owl="http://www.w3.org/2002/07/owl#"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema#"
   xmlns:rdfs="http://www.w3.org/2000/01/rdf-schema#"
   xmlns:rdf="http://www.w3.org/1999/02/22-rdf-syntax-ns#">
```

```
<owl:Class rdf:about="&preservation-case;ContentSet"/>
<owl:Class rdf:about="&preservation-case;PreservationCase"/> <preservation-case:
    hasObjective rdf:resource="CompressionTypeMustBeNone"/>


<org:Organization rdf:about="demo_organization">
   <rdf:type rdf:resource="&owl;NamedIndividual"/>
   <org:identifier>DEMO organization</org:identifier>
</org:Organization>

<owl:NamedIndividual rdf:about="100years_photos">
   <rdf:type rdf:resource="&preservation-case;ContentSet"/>

</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="100years_photos_scenario">
   <rdf:type rdf:resource="&preservation-case;PreservationCase"/>
   <skos:prefLabel>100 years photos</skos:prefLabel>
   <preservation-case:hasContentSet rdf:resource="100years_photos"/>
   <preservation-case:hasUserCommunity rdf:resource="public"/>
   <preservation-case:hasObjective rdf:resource="ImageDistanceSSIM"/>
   <preservation-case:hasObjective rdf:resource="ImageWidthMustBeUnchanged"/>
   <preservation-case:hasObjective rdf:resource="ImageHeightMustBeUnchanged"/>
   <preservation-case:hasObjective rdf:resource="ProducerMetadataRetained"/>
   <preservation-case:hasObjective rdf:resource="DateandTimeMetadataRetained"/>
   <preservation-case:hasObjective rdf:resource="CaptureDeviceMetadataRetained"/>
   <preservation-case:hasObjective rdf:resource="ExifAperturevalueRetained"/>
   <preservation-case:hasObjective rdf:resource="NumberOfToolsMustBeGT0"/>
   <preservation-case:hasObjective rdf:resource="
       FormatDocumentationAvailabilityShouldBeYesFree"/>
   <preservation-case:hasObjective rdf:resource="FormatShouldBeInternationalStandard"/>
   <preservation-case:hasObjective rdf:resource="ActionSoftwareLicenceMustBeOpenSource"
       />
   <preservation-case:hasObjective rdf:resource="SoftwareLicenseCosts"/>
   <preservation-case:hasObjective rdf:resource="ElapsedTimePerObject"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="ImageDistanceSSIM">
   <rdf:type rdf:resource="&control-policy;AuthenticityObjective"/>
   <skos:prefLabel>Image distance SSIM must be greater than 0.98</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;double">0.98</control-policy:value>
   <control-policy:measure rdf:resource="&measures;1"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
   <control-policy:qualifier rdf:resource="&qualifiers;GT"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="ImageWidthMustBeUnchanged">
   <rdf:type rdf:resource="&control-policy;AuthenticityObjective"/>
   <skos:prefLabel>Image width must be unchanged</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;boolean">true</control-policy:value>
   <control-policy:measure rdf:resource="&measures;51"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="ImageHeightMustBeUnchanged">
   <rdf:type rdf:resource="&control-policy;AuthenticityObjective"/>
   <skos:prefLabel>Image height must be unchanged</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;boolean">true</control-policy:value>
   <control-policy:measure rdf:resource="&measures;53"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="ProducerMetadataRetained">
   <rdf:type rdf:resource="&control-policy;AuthenticityObjective"/>
   <skos:prefLabel>Producer metadata element retained</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;boolean">true</control-policy:value>
   <control-policy:measure rdf:resource="&measures;296"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="CaptureDeviceMetadataRetained">
   <rdf:type rdf:resource="&control-policy;AuthenticityObjective"/>
   <skos:prefLabel>Capture device metadata element retained</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;boolean">true</control-policy:value>
   <control-policy:measure rdf:resource="&measures;291"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="ExifAperturevalueRetained">
   <rdf:type rdf:resource="&control-policy;AuthenticityObjective"/>
   <skos:prefLabel>EXIF: aperture value retained</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;boolean">true</control-policy:value>
   <control-policy:measure rdf:resource="&measures;252"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="SoftwareLicenseCosts">
   <rdf:type rdf:resource="&control-policy;ActionObjective"/>
   <skos:prefLabel>Initial software licence costs of an action must be less than 1000
       EUR</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;double">1000</control-policy:value>
   <control-policy:measure rdf:resource="&measures;26"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
   <control-policy:qualifier rdf:resource="&qualifiers;LT"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="ElapsedTimePerObject">
   <rdf:type rdf:resource="&control-policy;ActionObjective"/>
   <skos:prefLabel>Elapsed time per object must be less than 10 seconds</skos:prefLabel
       >
   <control-policy:value rdf:datatype="&xsd;unsignedLong">10000</control-policy:value>
   <control-policy:measure rdf:resource="&measures;11"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
   <control-policy:qualifier rdf:resource="&qualifiers;LT"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="NumberOfToolsMustBeGT0">
   <rdf:type rdf:resource="&control-policy;FormatObjective"/>
   <skos:prefLabel>Number of tools greater 0</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;integer">0</control-policy:value>
   <control-policy:measure rdf:resource="&measures;141"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
   <control-policy:qualifier rdf:resource="&qualifiers;GT"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="FormatDocumentationAvailabilityShouldBeYesFree">
   <rdf:type rdf:resource="&control-policy;FormatObjective"/>
   <skos:prefLabel>Format Documentation freely available</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;string">yes-free</control-policy:value>
   <control-policy:measure rdf:resource="&measures;147"/>
   <control-policy:modality rdf:resource="&modalities;SHOULD"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="FormatShouldBeInternationalStandard">
   <rdf:type rdf:resource="&control-policy;FormatObjective"/>
   <skos:prefLabel>Format should be international standard</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;string">international standard</control-
       policy:value>
   <control-policy:measure rdf:resource="&measures;161"/>
   <control-policy:modality rdf:resource="&modalities;SHOULD"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="CompressionTypeMustBeNone">
   <rdf:type rdf:resource="&control-policy;FormatObjective"/>
   <skos:prefLabel>Compression type must be none</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;string">none</control-policy:value>
   <control-policy:measure rdf:resource="&measures;117"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
</owl:NamedIndividual>

<owl:NamedIndividual rdf:about="ActionSoftwareLicenceMustBeOpenSource">
   <rdf:type rdf:resource="&control-policy;ActionObjective"/>
   <skos:prefLabel>Action software licence must be open source</skos:prefLabel>
   <control-policy:value rdf:datatype="&xsd;string">openSource</control-policy:value>
   <control-policy:measure rdf:resource="&measures;175"/>
   <control-policy:modality rdf:resource="&modalities;MUST"/>
   <preservation-case:contentSetScope rdf:resource="100years_photos"/>
</owl:NamedIndividual>

<foaf:Group rdf:about="public">
   <rdf:type rdf:resource="&owl;NamedIndividual"/>
</foaf:Group>

<owl:NamedIndividual rdf:about="&modalities;MUST"/>
<owl:NamedIndividual rdf:about="&modalities;SHOULD"/>
</rdf:RDF>
```

Bibliography

[BDF$^+$12] Christoph Becker, Kresimir Duretec, Luis Faria, Miguel Ferreira, and Jose Carlos Ramalho. Preservation Watch : What to monitor and how. In *Proceedings of the 9th International Conference on Preservation of Digital Objects (IPRES 2012)*, pages 267–274, Toronto, Canada, 2012.

[BKG$^+$09] Christoph Becker, Hannes Kulovits, Mark Guttenbrunner, Stephan Strodl, Andreas Rauber, and Hans Hofman. Systematic planning for digital preservation: evaluating potential strategies and building preservation plans. *International Journal on Digital Libraries*, 10(4):133–157, December 2009.

[BKH08] Christoph Becker, Hannes Kulovits, and Hans Hofman. Plato : A Service Oriented Decision Support System for Preservation Planning. In *Joint Conference on Digital Libraries*, pages 367–370, 2008.

[BKK$^+$09] Christoph Becker, Hannes Kulovits, Michael Kraxner, Riccardo Gottardi, Andreas Rauber, and Randolph Welte. Adding quality-awareness to evaluate migration web-services and remote emulation for digital preservation. In Maristella Agosti, José Borbinha, Sarantos Kapidakis, Christos Papatheodorou, and Giannis Tsakonas, editors, *Research and Advanced Technology for Digital Libraries*, volume 5714 of *Lecture Notes in Computer Science*, pages 39–50. Springer Berlin Heidelberg, 2009.

[BKPR13] Christoph Becker, Michael Kraxner, Markus Plangg, and Andreas Rauber. Improving Decision Support for Software Component Selection through Systematic Cross-Referencing and Analysis of Multiple Decision Criteria. In *Proceedings of 46th Hawaii International Conference on System Sciences (HICSS)*, pages 1193–1202, Maui, HI, 2013.

[BR11] Christoph Becker and Andreas Rauber. Decision criteria in digital preservation: What to measure and how. *Journal of the American Society for Information Science and Technology*, 62(6):1009–1028, June 2011.

[Bro08] Adrian Brown. Developing Practical Approaches to Active Preservation. *International Journal of Digital Curation*, 2(1):3–11, 2008.

[BSWW08] N. Beagrie, N. Semple, P. Williams, and R. Wright. Digital Preservation Policies Study Part 1: Final Report. Technical report, Higher Education Funding Council for England, 2008.

[DKK+14] Kresimir Duretec, Artur Kulmukhametov, Michael Kraxner, Markus Plangg, Christoph. Becker, and Luis Faria. The scape preservation lifecycle. In *Proceedings of the IEEE/ACM Joint Conference on Digital Libraries (JCDL 2014)*, pages 425–426, Sept 2014.

[FBR07] Miguel Ferreira, Ana Alice Baptista, and José Carlos Ramalho. An intelligent decision support system for digital preservation. *International Journal on Digital Libraries*, 6:295–304, 2007.

[FPD+12] Luis Faria, Petar Petrov, Kresimir Duretec, Christoph Becker, Miguel Ferreira, and Jose Ramalho. Design and architecture of a novel preservation watch system. In *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, volume 7634 LNCS, pages 168–178, 2012.

[GBR08] Mark Guttenbrunner, Christoph Becker, and Andreas Rauber. Evaluating Strategies for the Preservation of Console Video Games. In *Proceedings of the Fifth International Conference on Preservation of Digital Objects (iPRES 2008)*, pages 115–121, London, UK, September 2008.

[HB11] Markus Hamm and Christoph Becker. Impact assessment of decision criteria in preservation planning. In *Proceedings of the 8th International Conference on Preservation of Digital Objects (IPRES 2011)*, Singapore, 2011.

[HBS+08] H. Hofman, C. Becker, S. Strodl, H. Kulovits, and A. Rauber. Preservation Plan Template. Technical report, The Planets project, 2008.

[HT11] Steve Hitchcock and David Tarrant. Characterising and Preserving Digital Repositories: File Format Profiles. *Ariadne*, (66), 2011.

[HWS+06] Duncan Hull, Katy Wolstencroft, Robert Stevens, Carole Goble, Mathew R Pocock, Peter Li, and Tom Oinn. Taverna: a tool for building and running workflows of services. *Nucleic Acids Research*, 34(Web Server issue):W729–W732, July 2006.

[ISO03] ISO. Open archival information system – Reference model (ISO 14721:2003), 2003.

[ISO10] ISO. Space data and Information Transfer Systems - Audit and Certification of Trustworthy Digital Repositories (ISO/DIS 16363), 2010.

[KBA13] Hannes Kulovits, Christoph Becker, and Bjarne Andersen. Scalable preservation decisions: A controlled case study. In *Proceeding of Archiving 2013*, pages 167 – 172, Washington, DC, US, 2013. P. Burns.

[KKP+13] Hannes Kulovits, Michael Kraxner, Markus Plangg, Christoph Becker, and Sean Bechhofer. Open Preservation Data : Controlled vocabularies and ontologies for preservation ecosystems. In *Proceedings of the 10th International Conference on Preservation of Digital Objects (IPRES 2013)*, Lisbon, 2013.

[KPD+13] Michael Kraxner, Markus Plangg, Kresimir Duretec, Christoph Becker, and Luis Faria. The SCAPE Planning and Watch suite Supporting the preservation lifecycle in repositories. In *Proceedings of the 10th International Conference on Preservation of Digital Objects (IPRES 2013)*, pages 2–5, 2013.

[KRK+09] Hannes (Vienna University of Technology) Kulovits, Andreas (Vienna University of Technology) Rauber, Anna (Bavarian State Library) Kugler, Markus (Bavarian State Library) Brantl, Tobias (Bavarian State Library) Beinert, and Astrid (Bavarian State Library) Schoger. From TIFF to JPEG 2000? Preservation planning at the Bavarian State Library using a collection of digitized 16th century printings. *D-Lib Magazine*, 15(11/12), 2009.

[LSL+02] K.-H. Lee, O Slattery, R Lu, X Tang, and V McCrary. The state of the art and practice in digital preservation. *Journal of Research of the National Institute of Standards and Technology*, 107(1):93–106, 2002.

[Obj08] Object Management Group. Semantics of Business Vocabulary and Business Rules (SBVR), 2008.

[Pet13] Petar Petrov. Content profiling for digital preservation. Master's thesis, Vienna University of Technology, 2013.

[RGS09] David De Roure, Carole Goble, and Robert Stevens. The Design and Realisation of the myExperiment Virtual Research Environment for Social Sharing of Workflows. *Future Generation Computer Systems*, 25:561–567, 2009.

[Ros10] David S. H. Rosenthal. Bit Preservation: A Solved Problem? *International Journal of Digital Curation*, 5(1):134–148, 2010.

[Rot95] J Rothenberg. Ensuring the Longevity of Digital Documents. *Scientific American*, 272, 1995.

[Rot99] J Rothenberg. *Avoiding Technological Quicksand: Finding a Viable Technical Foundation for Digital Preservation.* Council on Library and Information Resources, 1999.

[SBNR07] Stephan Strodl, Christoph Becker, Robert Neumayer, and Andreas Rauber. How to choose a digital preservation strategy: evaluating a preservation planning procedure. In *Proceedings of the 7th ACM/IEEE-CS Joint Conference on Digital Libraries (JCDL 2007)*, pages 29–38. ACM Press, 2007.

[Sch12] Rainer Schmidt. An Architectural Overview of the SCAPE Preservation Platform. In *Proceedings of the 9th International Conference on Preservation of Digital Objects (IPRES 2012)*, Toronto, Canada, 2012.

[SM07] MacKenzie Smith and Reagan W Moore. Digital archive policies and trusted digital repositories. 2007.

[THC11] David Tarrant, Steve Hitchcock, and Les Carr. Where the Semantic Web and Web 2.0 Meet Format Risk Management: P2 Registry. *The International Journal of Digital Curation*, 1(6):165–182, 2011.

[Web03] Colin Webb. *Guidelines for the preservation of digital heritage.* Information Society Division United Nations Educational, Scientific and Cultural Organization (UNESCO) - National Library of Australia, March 2003.