University of BRISTOL

Waddington, S., Tonkin, E. L., Gill, A., Carter, A., Biermann, J., Eggers, A., ... Hedges, M. (Ed.) (2015, Oct 28). PERICLES WP5 D5.2 Basic tools for ecosystem management. *PERICLES Consortium*.

Publisher's PDF, also known as Version of record

Link to publication record in Explore Bristol Research
PDF-document

**University of Bristol - Explore Bristol Research**
**General rights**

PERICLES - Promoting and Enhancing Reuse of Information throughout the Content Lifecycle taking account of Evolving Semantics

[Digital Preservation]

## DELIVERABLE 5.2

### Basic tools for Digital Ecosystem management

Pericles
FP7 Digital Preservation

GRANT AGREEMENT: 601138

SCHEME FP7 ICT 2011.4.3

Start date of project: 1 February 2013

Duration: 48 months

| Project co-funded by the European Commission within the Seventh Framework Programme (2007-2013) | | |
|---|---|---|
| | **Dissemination level** | |
| **PU** | PUBLIC | X |
| **PP** | Restricted to other PROGRAMME PARTICIPANTS (including the Commission Services) | |
| **RE** | RESTRICTED to a group specified by the consortium (including the Commission Services) | |
| **CO** | CONFIDENTIAL only for members of the consortium (including the Commission Services) | |

# Revision History

| V # | Date | Description / Reason of change | Author |
|---|---|---|---|
| V0.89 | 21.09.15 | Internal draft | JB (UGOE) |
| V0.91 | 22.9.15 | Second internal draft | FC (ULIV) |
| V0.92 | 23.09.15 | Third internal draft. Merge into 5.2. | AC (UEDIN) |
| V0.93 | 23.09.15 | Merges, minor corrections, formatting | JB (UGOE) |
| V0.94 | 24.09.15 | Overall corrections, appraisal section | SW (KCL) |
| V0.95-0.97 | 29.09.15 | Integration of internal review comments | JB (UGOE) |
| V0.99 | 26.10.15 | Integration of changes on all chapters | JB (UGOE) |
| V1.0 | 28.10.15 | Final version for submission | CS (KCL) |

# Authors and Contributors

## *Authors*

| Partner | Name |
|---|---|
| KCL | Simon Waddington |
| KCL | Emma Tonkin |
| KCL | Alastair Gill |
| UEDIN | Adam Carter |
| UGOE | Johannes Biermann |
| UGOE | Anna Eggers |
| ULIV | Fabio Corubolo |
| ULIV | Jerome Fuselier |
| ULIV | Paul Watry |

## *Contributors*

| Partner | Name |
|---|---|
| XEROX | Jean-Yves Vion-Dury |
| UEDIN | Rob Baxter |
| KCL | Mark Hedges |

# Table of Contents

# List of Tables

# List of Figures

# Glossary

| Abbreviation / Acronym | Meaning |
|---|---|
| **BPMN** | Business Process Model and Notation. A graphical language for describing processes. |
| **Cassandra** | A distributed database system which is part of the Apache foundation.[1] |
| **CDMI** | Cloud Data Management Interface is a protocol for accessing cloud storage. |
| **CEPH** | CEPH is a distributed file system. |
| **Content-based (or intellectual) appraisal** | Acquisition and retention decisions or assignment of value based on the content of the digital entities themselves. |
| **CQL** | Query language to access the Cassandra database. |
| **DBA** | Digital-born Archive |
| **Digital Ecosystem (DE)** | Network of technical systems, communities, digital objects, processes, policies, and the relations and interactions between them. This is the object of interest that is modelled with the Digital Ecosystem Model ontology. |
| **Digital ecosystem management** | Control layer to provide support and manage change in the digital ecosystem and its entities. In the scope of this task, the QA methods are supporting the validation of changes in the digital ecosystem with respect to policies and high value digital media. |
| **Digital Ecosystem Model (DEM)** | Ontology developed by the PERICLES project that allows to model Digital Ecosystems: technical systems, processes, digital objects, policies and users to answer and simulate change related questions. |
| **Digital Object** | "Digital objects (or digital materials) refer to any item that is available digitally." (JISC, "Definition of Digital Object") |
| **DoW** | Description of Work |
| **ERMR** | Entity Repository Model Repository this refers to the T5.1 component. |
| **iRODS** | The Integrated Rule-Oriented Data System (iRODS) is an open source data management software that virtualizes data storage resources. The application can be used for data management infrastructure building. |
| **LDAP** | Lightweight Directory Access Protocol - standard protocol for |

---

[1] http://cassandra.apache.org/

| | |
|---|---|
| | distributed directories |
| **LRMS** | Linked Resource Model Service |
| **Policy** | Used in very diverse situations both in English, and in IT. A policy is a plan that defines the desired state inside an ecosystem. A policy describes the 'what' (guidelines) and not the 'how' (implementation). Policies can be described in varying degrees in natural language or in a formal language. Policies can also be used to represent the legal requirements and aspects of an ecosystem. |
| **Quality Assurance (QA)** | *"Program for the systematic monitoring and evaluation of the various aspects of a project, service, or facility to ensure that standards of quality are being met"* (Webster) |
| **RDF** | Resource Description Framework. A versatile data model in which assertions are expressed as *subject-predicate-object* triples. |
| **ReAL** | The Resource action language describes transformative actions on RDF based models. Enables rule functionality on ontology. |
| **REST** | Representational State Transfer. A design style for networked applications, usually implemented with HTTP. |
| **SBA** | Software-based artwork |
| **SPARQL** | SPARQL Protocol and RDF Query Language. An RDF query language |
| **Technical appraisal** | Decisions based on the feasibility of preserving the digital objects. This involves determining whether digital objects can be maintained in a reusable form and in particular takes into account obsolescence of software, formats and policies. |
| **Unit Test** | Technique that originates from software engineering for modular testing of source code |
| **VBA** | Video-based artwork |

# 1.  Executive Summary

This deliverable, *Basic Tools for Digital Ecosystem Management* describes the current state of the WP5 digital ecosystem (DE) research and tools developed for the PERICLES project, together with the concepts, models and software associated with them. The developed tools can be used independently or in combination with test scenarios, which will be part of the WP6 test beds.

The ecosystem tools cover a broad area of topics:

- The PERICLES Entity Registry Model Repository (ERMR) (T5.1 and T5.2) is responsible for storing models and registering entities. The component enables an evolving digital ecosystem where entities can undergo a managed change with the help of quality assurance and risk analysis functionality. The possibility to define policies and triggers on ERMR (T5.2) supports managed change. The ERMR is used by the integrated test bed as a central registry for entity descriptions and storage service for models.

- The development of quality assurance methods (T5.3) enables to add quality assurance criteria and verification to the models: this is an approach to verify that the entities inside the model comply with defined criteria. It is not about validating properties of digital objects, but about adding and embedding QA functionality at the model level.

- Appraisal (T5.4) is concerned with two issues: what (for an archive) should be acquired and what should be retained. It is a traditional discipline for an archive and mostly a manual process. The aim of this task is to specify criteria that can be automated and to provide associated methods and tools. The task deals with both the value of the content and technical issues in its preservation.

- The current development of the Digital Ecosystem Model has been moved from WP5 to WP3 (T3.5.2). However, since it has relevance for the quality assurance task, provides a software component for model instantiation and is one of the models that will be stored in the ERMR, we have decided to include a brief description in this deliverable.

# 2. Introduction & Rationale

This deliverable describes the current state of development of the tasks in WP5 providing methods and tools for the management of digital ecosystems. *Digital ecosystem* is a term chosen to reflect collections of interdependent entities, to which the model-driven preservation approach being developed in this project can be applied. Managing digital ecosystems is currently repetitive and time-consuming work, requiring specialised knowledge and continuous monitoring of the state to guarantee correct functioning with respect to the policies and guidelines, management principles and decisions.

The tasks described in this deliverable are: T5.1 entity registry, T5.2 process, policy and process infrastructure, T5.3 quality assurance and T5.4 support functionality for appraisal processes. In addition, the current progress of the task T3.5.2 is briefly described. This task works on a programmatic approach to work with the Digital Ecosystem Model ontology and therefore falls under the category of digital ecosystem management tools.

To make the role of the WP5 tasks with regard to the other project developments clear, the relation to the PERICLES components will be described in this chapter. There is also a dedicated task T3.5.3 for developing the functional architecture, which will include a combination of the project tools and research output, while the focus here is on the WP5 tasks.

## 2.1. Context of this Deliverable Production

### 2.1.1. What to expect from this Document

PERICLES is developing a model-driven preservation approach. During this project several models, together with associated concepts and components are being produced. All of the WP5 tasks will continue until M46, and Task 5.2 has just started, so this document is a progress report on the ecosystem tools and the future outlook. WP5 works on these tasks:

- T5.1.1 "Registries for digital ecosystem management":
  the entity registry model repository (ERMR) is responsible for registering entities and storing the different models. It provides access to the information via a query interface.
- T5.1.2 "Policy editor": implements a policy editor that allows changing policies on the entity registry and entity store. As this task has just begun, it will be reported on in the next deliverable D5.3.
- T5.2 "Develop processes for digital ecosystems":
  adds event trigger and policies (rules) on ERMR to execute processes. In addition, this task populates the store with some common preservation processes to react on an event (e.g. model update).
- T5.3 "Develop quality assurance methods for digital ecosystem management and semantic evolution":
  enables us to define and validate quality assurance criteria that can be integrated into a

model to enable validation, for example on the digital ecosystem model.

- T5.4 "Support functionality for appraisal processes":
  concerns appraisal processes and risk analysis, which of the data needs to be kept, what are the associated preservation risks.

### 2.1.2. Relation to other work packages and output

Each task of WP5 consists of a research part and practical part. The practical part will feed into the test bed, while some research parts might go into the T3.5.2 digital ecosystem model. In particular the tasks will produce the following components:

- ERMR T5.1 and T5.2: Will be a part of the test bed infrastructure. It is the central registry and model store of the test bed. The purpose of ERMR on the test bed is to provide a central registration of entities and models for the test scenarios. The T5.2.1 component works on top of ERMR. It allows the definition of policies in form of rules to react to operations that are being performed involving the ERMR. The rules enable the execution of processes. A subset of common long-term data management policies and processes will be selected for the definition of example rules and processes that demonstrate that ERMR can support the necessary operations from this area.

- T5.3 quality assurance develops a policy model and concept that allows the integration of QA into models. As an example for the integration of QA into a model, the definitions of the policy entity will be integrated into the T3.5.2 digital ecosystem model ontology. This allows us to provide test cases for WP6 that demonstrate different aspects of this task.

- T5.4 is a research task about technical and content-based appraisal. It examines at the WP2 case studies as well as publicly available collection policies to identify the requirements of appraisal and risk analysis. The research output will be a set of methods, models and processes for evaluating appraisal criteria. Prototypes and test scenarios will demonstrate the work in a set of selected examples.

- T3.5.2 Digital Ecosystem Model: The output will be an ontology together with a detailed description and some examples. It is dependent on the WP3 LRM upper ontology. Outputs from T5.3 task will be integrated and if applicable, also from T5.4.

## 2.2. Document Structure

The document structure reflects the order of the WP5 task as listed of the DoW.

The first chapter will present the progress on the Entity registry in T5.1, followed by the process development for digital ecosystems in T5.2, then describe work done on quality assurance methods in T5.3 and finally progress with regard to appraisal processes in T5.4. The last part reports briefly about the WP3 T3.5.2 ecosystem model.

# 3. Entity Registry Model Repository (T5.1)

The primary objective of the task is to provide a registry for entities and a store for models. This middleware application is called the "Entity Registry Model Repository (ERMR). It is a uniform application that provides a user interface and a machine processable interface (API). The registration part of ERMR will be used to coordinate the management of externally stored data and keep further information about entities that are referenced in the models. The resulting middleware is intended to support services based on the PERICLES models and interaction with the Linked Resource Model service, applying the project's change management functions to the test bed data and metadata. It functions as a central component of the test bed: all services and data pass through the ERMR.

Using the ERMR, it will be possible to activate several LRM services dedicated to model various domains - for example, tracking or controlling the evolution of the target ecosystem; tracking the evolution of LRM ontologies; applying the LRM change management language (ReAL), and so forth.

This chapter describes the evolution of the ERMR as a scalable Cassandra-based system, with a rule engine that can trigger events on distributed data stores and will support policy management requirements. The tool incorporates a query mechanism that supports multi-tiered access control and a logging system that supports authentication, and auditable logs. The ERMR is implemented in terms of a standardised, extensible framework that could in future be extended further to support an integrated data management application, such as an archive.

The T5.1 task relates to the D3.4 deliverable (ReAL operated by the LRM service) and the D6.4 deliverable (test-bed implementation), discussed in other deliverables, along with Task T5.3 (Quality Assurance) and Task T5.4 (Support for Appraisal Processes), described below. Whereas Task T5.1 sets out the repository model; it forms the basis of Task T5.2, which sets out the process infrastructure.

## 3.1 Core functionality of ERMR

The present section discusses the scope of the ERMR, which constitutes policy-based middleware that can be used to negotiate services between the LRM service and the test bed data. In a nutshell:

- The ERMR will act as a persistent store for the models and registration information of entities. This is important for the test bed, because ERMR holds the required information for the test scenarios. Further, it is also important for the LRM service because the LRMS itself does not maintain a persistent store of its outputs.
- The ERMR will support a query mechanism to store and retrieve the data. To retrieve or store an object, the full path in the container hierarchy must be specified. Alternatively, retrieval may be via the repository assigned unique identifier, as per the CDMI specification. It is also possible to locate objects by specifying metadata names or values in the URI query field. Triples may be added through an HTTP POST command. The triple store may be queried by issuing an HTTP GET on the triple store objects with the SPARQL query specified in the URI.

- The ERMR provides distributed and extensible store capabilities, including object stores, row stores, and fact stores; with an easy to use trigger / rules system. The object store (or blob store) has simplified semantics with corresponding increases in performance, reliability, transparency, and ease of use. The fact store (or triple store) store individual triples that can be inference over to extract relationships that emerge from all the facts. Row stores (tables) group related records, so that they may be selected, sorted, and otherwise manipulated in aggregate, without having to load and unpack each individual record, as would be the case with XML.

- The ERMR communicates with the LRM service. This will allow inferencing, developed as part of the project. This is important because it can apply LRM services to many data-intensive problems including information discovery, entity resolution and information extraction of the models.

- The ERMR supports data sharing, or interoperability between external data stores. This "virtualisation" capability is important because of the need to operate across radically different types of storage and processing technologies; and for the ability of users to access and share data. The actual data (e.g. the blob of a digital object) is not held in the ERMR.

- The ERMR supports the generation and maintenance of audit trails for specific operations or events. This is important in order to trace the history of the operations, and to prove that all operations on the digital entities were performed by authorised users, including the update of the authenticity metadata itself. We expect PERICLES inspired architectures to be used across different communities of practice, operating across security boundaries. This will require a policy framework for the description and analysis of security policies. It will also require implementation of interoperability mechanisms used to support interoperability across identity management systems and authentication systems, based on pluggable authentication modules.

- With the audit trails ERMR supports data management policies and processes that can be used to demonstrate the management of the lifecycle. This is important for implementing lifecycle management policies as computer-actionable rules, which can be applied across different storage technologies. The policy framework will support triggering of quality assurance processes.

- The ERMR supports authentication as part of its ability to track the identity of users working in a high-security environment. This is important given confidentiality and data protection requirements.

Described at a higher level: The "loosely coupled" ERMR and LRM tools will support three core advances in the project:

- At a basic level, providing modification of the store and making for Resource Description Framework (RDF) information;

- at an intermediate level, supporting some LRM specific services (to be defined); and

- at an upper level, signalling changes occurring in the ecosystem (creation, deletion, update), which may be used to perform some transformative actions, as required (T5.2.1).

# 3.1. State of the Art

The project considers the preservation of digital objects as part of continually evolving ecosystems, in which models as well as digital objects are preserved. This view represents an evolution in conceptual thinking, which has its roots in the preservation and lifecycle management approaches.

The technology supporting this requires a step forward from many conventional preservation approaches, in that it assumes that distributed data will be managed locally, and not migrated to a central "archive" repository for long-term curation. This has prompted the development of adaptive middleware technology to provide suitable "virtualisation" technologies, such that services can be applied across different storage technologies and infrastructures, including potentially future technologies as yet unknown, and legacy technologies no longer widely supported or documented.

There are very few available systems possessing this capability and none that we are aware of, which can be configured to support the relational approach of the PERICLES LRM Services. However, we are not working in isolation and a brief overview of related initiatives present some valuable insight into how such an approach may be crafted; and how such an approach relates to significant work in the field. Overall, we see this development as a contribution to a better understanding of preservation, whichever approach is adopted.

Our initial investigation suggests how different communities develop technologies to support the long-term curation of data. These include the data grid community (through the Global Grid Forum working groups), the archive community (through the application of prototype preservation environments), and the digital library community (through initiatives such as METS, supporting the discovery and access of materials).

Each community has focused on an aspect of the problem, or equivalently a subset of the processes required for long-term curation of data. For PERICLES, the development effort informing this task (ERMR) took into account research groups focusing on different areas in order to get an overview of requirements, and to evolve an approach that represents the state of the art in terms of digital curation. The list below is by no means exhaustive, but covers some of the more notable efforts. These include, among others:

- The Integrated Rule Oriented Data System (iRODS), which remains the closest system to implementing all components and used in multiple EU projects (www.irods.org).
- The Storage Resource Broker (SRB) from Nirvana, which provides the data and trust virtualisation needed for infrastructure independence, but lacks management virtualisation (the ability to express management policies). (www.ga.com/nirvana).
- The DSPACE repository software, which provides standard services for ingestion and access (www.dspace.org)
- The Fedora digital library system, which is considered as middleware that can be used to implement a preservation environment (www.fedora-commons.org)
- Semantic grid technologies, for managing reasoning on attributes inferred about a collection through use of ontologies (www.mygrid.org.uk).
- The PLANETS project focused on representation information and management policies (www.planets-project.eu).

- Related work in EU integrated projects (PrestoPrime) has resulted in the development of generic services needed for the manipulation of structured information. (www.prestoprime.org).

It will be noted that the most relevant approaches, including iRODS and SRB, have developed strategies to create virtual data archives where the data are seamlessly preserved and curated (in the case of iRODS, with policy-based rules). The applicability of such approaches to the management of research data is well known and forms a key building block in understanding how we can better extend these approaches to meet the data continuum.

## Justification of Technology Choice

Can the current generation of technologies be tailored to achieve the goals of the project? Or will it be necessary to develop new technologies based on the need to support transactional services, which require distributed access to tables, fact stores (triples), and object stores?

To inform our choice, we implemented two architectures for the ERMR. The first was developed in Year 1: this was based on the iRODS system, developing extensions that support the python scripting language ("pyrods") for executing policies / rules. This was implemented on a server at UGOE.

The second, developed in Years 2-3, was based on the Apache Cassandra open source distributed data system. This version was developed specifically to support the linked and relational database management capabilities required for LRM services. This has also been implemented on a server at UGOE.

While there are advantages to each approach, there are some distinguishing features that favour the Apache Cassandra system, at least for this project.

- A current limitation of the iRODS system is that it consists of an object store only; such an architecture might not easily support the RDF based models required for the LRM services (requiring a fact store).
- A second limitation is represented by the syntax of the iRODS rule system, which may not easily extend to supporting policies as envisaged for this project. The Cassandra-based ERMR application operates a "trigger" system that allows actions in any scripting language, described in Section 5.2 below.

We maintain an open mind given that these technologies evolve and adapt to new needs. An interesting prospect is whether we can migrate data, policies, and audit trails from one system to another without loss of context. This would constitute a "proof" that collections can be migrated from one technology to another, under a common set of management policies and procedures, without changes to properties. This represents validation of the research undertaken by Reagan Moore, Richard Marciano, and other colleagues during the past twenty years, which will have great relevance to the "non-custodial" continuum approach.

# 3.2. Architecture Overview

This section presents an overview of the Cassandra-based system, which we are now using as the principal basis to deliver "change management functionalities" described in Deliverable D3.3 to external clients. We express this system in terms of *adaptive middleware* architecture that is sufficiently flexible to support the LRM Services across a range of requirements, use cases, and user communities.

This diagram presents the architecture of the ERMR:
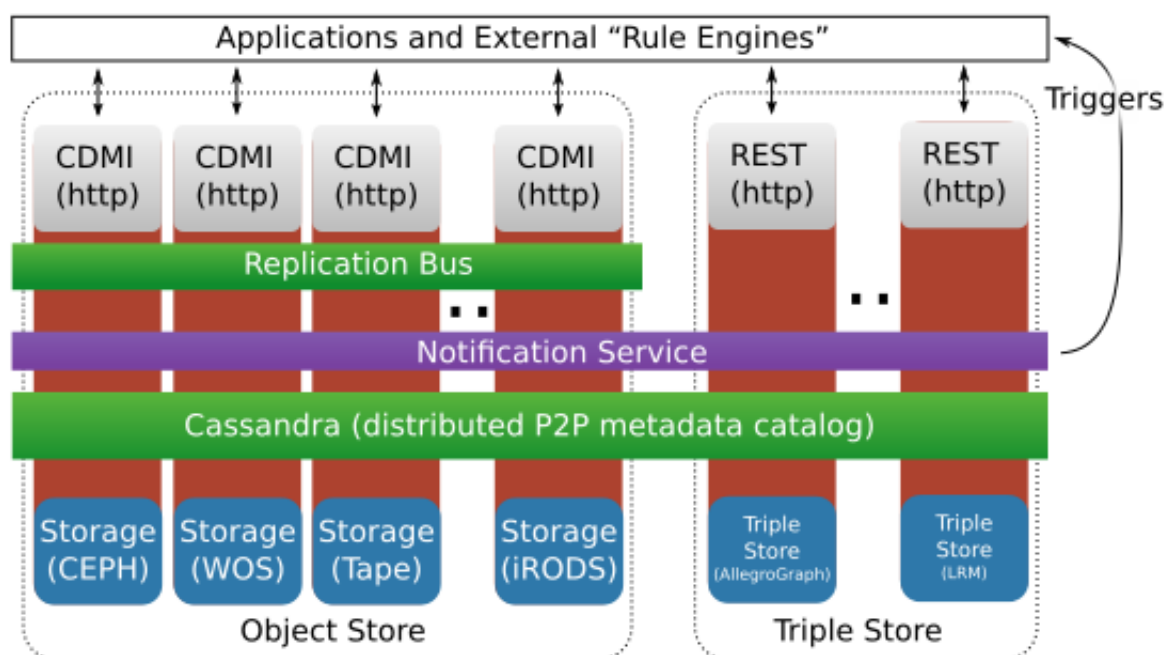


Figure 1: Architecture of the PERICLES Entity Registry

## 3.2.1. Implementation

A more detailed description of the technology is presented in the Appendix 2. A summary of major features, as set out in the appendix, includes:

- *Storage and content:* An explanation of the storage architecture, which is set out in terms of its client/server model, rule system, and metadata catalogue.
- *Identification system:* The method by which the ERMR assigns unique identifiers and accessible locations in which objects are located; and to synchronise the different representations of entities within the ecosystem.
- *Query mechanism:* An explanation of the standardised query mechanisms (e.g., SPARQL and REST API) that will support multi-tiered access control, and different views of stored data.
- *Response format:* The capacity for the ERMR to output models in a range of interpretable formats.
- *Notification (or trigger) functions:* An explanation of a trigger-based event system, with generic language support for actions, with the implicit co-location of action to data.

- *Logging/auditing:* Including audit able logs, authentication, and access control.

The following section discusses in more detail the technical components developed to support the architecture.  Further information is included in the Appendix 2:

- The server software, installed at one or more locations, uses state information to record all metadata attributes that are needed about a file, including the name of the file, the location of the file, the owner of the file, a file checksum, and data expiration data, and other attributes.

- The different components are accessible with a RESTful interface and every modification of the state of the registry is sent to a messaging queue so other components can react to changes.

- The Digital Objects store provides a metadata catalogue stored in a row-based database. It is responsible for the virtualization of different storage technologies.

- ERMR supports a substantial subset of the Cloud Data Management Interface (CDMI) API. CDMI specifies a rich set of operations in a unified and consistent manner to access containers, objects in containers. The CDMI API can be used to manipulate Digital Objects. It can organise objects within collections and associate metadata and ACL to objects.

- A rule engine ("listener") includes support for rule actions in any scripting language (e.g. Python) and RFC 5424 protocol (syslogNG) is used for patterns and trigger mechanisms. The rule engine will be used to automate the enforcement of policies as required for the use cases. The ERMR supports data management policies at a micro level (e.g. replication); and at a macro level (e.g. policies used in the ecosystem models to manage the evolution of the model).

- Apache Cassandra is used as a massively scalable database, well suited to distributed repositories, with very fine-grained access control. It will support user-defined tables via the Cassandra Query Language (CQL).

- User groups can be given Read Only or Read Write access to specific collections; and the catalogue distribution is automatic. Object location can be finely tuned using intuitive policy replication

- Nodes can be added and removed at will, allowing casual expansion and contraction of the registry with automatic rebalancing of both catalogue and storage resources.

- Object location can be finely tuned using intuitive policy replication. Parallelism, resilience, and scalability are achieved via HTTP redirections, which require no "special" mechanisms.

- ERMR uses Lightweight Directory Access Protocol (LDAP) to authenticate users. This is an open industry standard application protocol for accessing and attaining distributed directory services for an Internet Protocol (IP) network. A common use of LDAP is to provide a simple sign-on, where one password for a user is shared between many services.

- ERMR is controlled using standardised ACL (access control lists), which can be applied at any level and will apply to the sub-tree in the same way as storage policy.

Wherever possible, existing and widely used standards are used to leverage the Internet developer community's efforts to provide on-going support and reliability.

# 3.3. Outlook

We have presented an extensible and resilient adaptive middleware architecture that is based on policy-oriented data management, for supporting the data continuum model and management processes across different heterogeneous resources. This has been implemented as a "Reference Implementation" that is capable of being extended to production. The interface described, supporting a range of client applications, is based on the Apache Cassandra database platform, which provides resilience and scalability, as well as promoting sustainability.

The reference implementation supports the requirements set out in the description of work. It is designed to track entities and relationships in a digital ecosystem; and the policy-based data management framework that is needed to support application of processes. The policy-based application is capable of implementing interoperability mechanisms needed to link the LRMS with the underlying data stores. The architecture supports the management of data curation in a non-custodial environment. It accomplishes these requirements through technical interoperability, policy, and end user usage requirements. The result is used to track evolution of management policies, in accordance with the LRM Services. These include policy requirements that represent sets of processes or workflows as developed in response to the project's ecosystem and LRM modelling. Populating the registry is a task that is covered in the D6.4 deliverable.

At the time of writing, the ERMR services all operate within the "core" of the software. The next step will be to investigate what services might be extended beyond the core, using the rule engine. This should increase the power and scope of the software's ability to apply curation management policies in a distributed infrastructure: relevant capabilities may include federation, versioning, Hierarchical Storage Management (HSM) migration, processing services, and so forth, as required. The application is not limited in its extensibility, and may be tuned to meet the evolving needs of different communities.

The project forms the basis for future use based on digital library services, to support large-scale publication, indexing, and curation. Collectively, we will wish to leverage the development of interoperability mechanisms and generic services that allow in future re-use of data through mapping to a new context; this will include the manipulation of descriptive metadata.

# 4. Develop processes for digital ecosystems (T5.2)

This task is a requirement for the set of *policy-based data management functions* that can be executed by the ERMR tool in support of the ecosystem model and the LRM. As described, the transformation of data management policies into computer-actionable rules is an essential capability that underpins the PERICLES infrastructure. The assumption is that policies that are used for preservation form a continuum with earlier stages of the data life cycle, and are required for future use or repurposing of collections. We are therefore interested in the possibility of *evolving policies*, as enabled through the Linked Resource and Digital Ecosystem Models. The policies for the LRM service relate to ecosystem management; this may be distinguished from the ERMR policies that can register data management procedures.

The ERMR, described above, provides the operational functionality to support this task, using a *rule engine* that can trigger events. The rule engine represents a system in which an *action* (create, update, delete an object, model or its metadata) will cause a *procedure* to run.

The ERMR is able to execute conditions and actions through *triggers*. Conceptually, an event (such as a deposit) will *trigger* the evaluation of a condition that determines whether or not an *action* executes. A *condition* will determine whether the event is a candidate for action (such as the addition of accounting information when a file is created); or whether no action is required.

This capability is supported in ERMR by a *listener*, which will compare notification to a set of patterns and, when matched, will execute the appropriate procedure. The selection of policies and procedures will support the types of services that are required. The patterns are stored as *metadata*, which are updatable, as the use of the data store may evolve (e.g. as data collections are repurposed). When an action is required, the ERMR will be able to execute scripts that will execute procedures or services within a workflow.

The approach can be used to develop a range of services, such as follows:

- Automatically dispatch a copy to a remote system
- Normalise a deposited object into a "preservation" format, e.g. by creating PDFs
- Populate metadata using data retrieved from, for example, a corporate database of users
- Notification by email of significant events
- Insertion of retrieval data into a search index
- Involve the LRM service for evaluating the change
- Execute QA methods that are defined on the models (T5.3)

## 4.1. Example Scenario

The types of services and workflows supported by the "listener" are to be set out in Deliverable D6.4, at which point the services will be scripted and implemented. As deployed on the PERICLES test bed (described in D6.4), the LRM service is "hidden" from running ecosystem components behind the

ERMR interfaces; communication between ecosystem components in the test bed and the LRM service are brokered transparently by the ERMR.

An example of an LRM service consists of the following: A client will ask for a change impact evaluation from the ERMR, which will inform the LRM service about changes occurring in the ecosystem (creation, deletion, update). It will signal these as events, which may trigger reactions from the LRM service. Depending on the configuration of the LRM services (encoded through dedicated triples and ReAL specifications), those events can lead to internal modifications and/or may also trigger calls to external services (most likely again through the ERMR) in order to perform some transformative actions (e.g., launching a command to verify the validity of an XML file, computing a digest, etc).

# 4.2. Outlook

The flexibility of this approach leads to the concept of "rule packs". Given that many, or most, features or extensions will be implemented by the use of rule sets, we can develop "libraries" of policies or rules needed to perform the management of data, which can be tuned to meet the needs of different user communities. A rule set can also originate from common long-term preservation guidelines and policies, which for the project reference the LRM Services and ecosystem models; however, others may be developed, depending on need or collection use. The ERMR supports the technology, through its policy-based framework, which will enable different communities to control their use of a shared data collection. The ERMR could potentially support a "shared repository" that is distributed across different communities, in which data management or preservation policies may be automated and invoked. A future goal would be the development of standard rule packs, or policy sets, that can be modified for use as required. The expectation is that these can be analysed for generic infrastructure that is common across both science and arts disciplines. The hope is that this will provide the foundation for different communities to build upon the lessons learned; this will contribute to a reusable infrastructure, promoting best practice at minimal overhead.

# 5. Quality assurance (T5.3)

## 5.1. Task definition and scoping

This task's main objective is to define a series of Quality Assurance (QA) criteria for the entities of evolving ecosystems, in particular policies, processes, complex digital media objects, semantics and user communities. This will allow managing change in the ecosystem by validating its entities, detecting conflicts and keeping trace of its evolution through time. This task will make use of the existing ecosystem definitions, models and entities illustrated in Chapter 7, the LRM model (WP3) and related ontologies (WP2) developed in PERICLES.

There is an important distinction in the scope of this task: we aim to provide QA of policies, which shouldn't be confused with policies for QA. Our methods aim to validate the correct application of policies to the ecosystem. When change happens, the approach will ensure that policies are still correctly implemented. This is different to evaluating QA criteria on a digital object. Instead of operating on specific digital object related issues, such as validating format migration, this task works on integrating the QA approaches into the models. We consider this an important task: it will allow tracing the correct application of the higher-level policies (guidelines, principles, constraints) in the concrete ecosystem implementation.

Policies will be expressed at different levels, using the policy model and derivation method described in paragraphs 5.3 and 5.4, which are integrated in the Ecosystem Model itself. We support the QA of policies by defining criteria and methods that can validate or measure the correct application of policies through processes, services and other ecosystem entities, so assuring that the implementation is respecting the principles defined in the high-level policies. The QA methods will in turn support the management of change in the ecosystem entities, such as change in policy, policy lifecycle, change in the processes implementing those policies, or change in other policy dependencies. These methods will allow statements to be made about the ecosystem and its consistency with respect to the entities in consideration; and recognise changes that can be problematic and require action.

### QA of policies and processes

We investigate how policy implementation in an ecosystem could be described by using the **LRM, ecosystem models and dependency concepts**, based on a formal description of the policy implementation from high level down to processes, services and rules. This task defines a set of QA criteria in order to support policy validation. Changes to policies or to other ecosystem entities will require validation of the policy compliance using the QA criteria. The quality for a policy will be the expression of its correct implementation in the ecosystem. When a Digital Object for example, is migrated between institutions the QA may allow evaluating if the digital preservation or general policies are still valid in the new/changed ecosystem.

In this task we are **not making any strong assumption on the format in which the policy is expressed**, be it natural language, or a structured format or formal language, nor we are imposing

any specific structure on the implementing processes. We are assuming that policies and processes in real systems will be implemented using a variety of techniques and we aim to develop a policy layer that can be applied on top of existing ecosystems. This assumption will allow the deployment of such QA methods in systems that are not built using only specific technologies or rule languages, making their adoption much simpler. It has the benefit of not imposing radical changes or restructuring of the existing architecture, which is in line with the principles expressed already in the deliverable D5.1 Initial Report on Preservation[2] Ecosystem Management. An external model for the policy entity will be discussed in paragraph 5.3. The model will be generic and does not only apply to digital preservation policies, as we consider that there will be overlap between preservation and general policies, and those naturally will need to coexist.

### Scenarios and Objectives

- Top-down: know what processes depend on a specific policy to validate them in case of policy changes.
- Bottom-up: know which policy depends on a particular ecosystem entity, so that any change related to that entity could trigger policy validation and QA methods. Knowing the policy-process graph will also help notice when the practice starts to deviate from the guidelines that will help update the policies or correct the practice.

### Importance of the task

In order to provide reliable management of the ecosystem, Quality Assurance (QA) will be a valid and important methodology to validate and guarantee coherence after changes to the different ecosystem entities. Change and obsolescence are frequent, thus the ecosystem is continuously evolving and requires constant monitoring to trigger QA that will be supported by our task.

# 5.2. State of the art on QA and change management

Quality management is an umbrella term that includes all disciplines that deal with quality inside an organisation. Quality means to ensure a certain desired output according to policies or well-defined processes. There are different fields involved in quality management, which are shortly presented in this section.

---

[2] Note that since its publication in July 2014, we have changed the terminology from preservation ecosystem to digital ecosystem to express that a distinct preservation management or system is not a pre-requirement for implementing the tools and approaches proposed by PERICLES.

## 5.2.1.  Quality management systems

These are methods that include the whole organisation to maintain and improve a desired level of quality. One example is the ISO 9000 standard series. It defines principles, responsibilities, and measurement throughout the realisation of a product including the service sector. It consists of several distinct quality topics and typically includes several of the following quality disciplines:

- **Quality policy**[3]: a high-level expression which kind of quality should be achieved and what are the main aims;
- **Quality planning**[4]**:** takes the quality policies and refines the high-level quality policies into objectives and requirements. There is a set timeframe for achieving these objectives and requirements;
- **Quality control**[5]**:** a process that checks and ensures that a product or service meets the defined standards of the organisation or a certain norm that should be applied for the item. It is an active process that is performed manually or automatically and verifies the output (product or service) against the standards (policies).
- **Quality improvement:** a recurring process task that takes all measurements into account. The measurements come from the planning, assurance and quality control. It gives feedback to the quality planning and assurance processes for improvement.

## 5.2.2.  Change management

While quality management methods and the different sub-topics deal with establishing and maintaining a defined set of quality standards, change management deals with controlled processes for introducing change. Change management is well known in two areas: In Information Technology in form of ITIL standard[6] and in economics.

- Change management in IT is about modifying an IT infrastructure in a controlled way. To perform a change the change management defines a whole procedure in a form of processes and manual workflows that describe what needs to be done and who is responsible. For example ITIL suggest creating a change request, classifying it, analysing the use, risks and costs, accepting and planning it, creating the documentation, information and evaluation about the change[7]. It ensures that a change to an IT infrastructure is performed in a structured and well-defined way.
- In economics change management is a discipline of introducing change into an organisation. In comparison to IT change management this change is not so strongly bound sequential and strict processes and procedures because the topic of a change can be broad. It roughly undergoes a set of transitions: identify that a change is necessary, clarification, planning, accomplishment, monitoring and maintenance. The change can apply to all parts of an organisation: strategy, products, personal, processes and many more fields[8].

---

3 http://www.businessdictionary.com/definition/quality-policy.html
4 http://www.businessdictionary.com/definition/quality-planning.html
5 http://whatis.techtarget.com/definition/quality-control-QC
6 https://en.wikipedia.org/wiki/ITIL
7 http://wiki.en.it-processmaps.com/index.php/Change_Management
8 http://www.financepractitioner.com/dictionary/change-management

There are also change management approaches in other fields like production processes, which have in common to introduce a change in a structured, controlled way.

## 5.2.3. Quality validation and testing

The quality control or testing is typically specific to the product or service that should be produced or provided. In the following section an overview of different areas is provided.

### Automated testing and production testing

Production testing performs test cases to validate a product, service or software against desired policies and requirements. This is done to guarantee a certain level of quality and to ensure that no defective product leaves the factory. This area has similar evaluation criteria to software testing. Some of the approaches are intended for automatic tests, e.g. dimensional accuracy, surface defects, automated testing in electronics with a test jig or a bed of nails. Other criteria need manual inspection, such as the quality control of finishing and cleaning up of residues from production.

### Software engineering

Testing is a stand-alone discipline for quality assurance in software engineering. There are many different test categories for software tests[9], e.g.:

- **Requirements and analysis testing:** This can include acceptance testing, prototyping, scenario testing, e.g. with user stories
- **Architecture/Design testing:** Model reviews, code proving, specification based testing
- **Code testing:** Black- and white box testing, unit testing, regression testing, code coverage, and other metric based quality assurance methods
- **System testing:** Operation and stress testing, function testing, installation testing, integration testing, security testing, performance testing
- **User testing:** Alpha/Beta phase, user acceptance testing, usability testing

**ISO/IEC/IEEE 29119 software testing standard**

The 29119 standard series contain a description of software testing procedures. It consists of five parts:

1. 29119-1 Concepts and definitions
2. 29119-2 Test processes
3. 29119-3 Test documentation draft
4. 29119-4 Keyword-Driven Testing
5. 29119-5 Test Techniques

It basically covers the different test methods described above together with information on how to structure the processes around testing and documentation, and replaces the older standards IEEE

---

9 categories takes from the FLOOT lifecycle http://www.ambysoft.com/unifiedprocess/aup11/html/test.html)

829 (Test Documentation), IEEE 1008 (Unit Testing) and BS 7925 (software testing glossary).

There is also the IEEE standard 1012-2012 which is about validation and verification (Standard for System and Software Verification and Validation). It describes the necessary verification and validation activities during software development and testing. It is a kind of checklist and activity description and not about details of verification.

**ISO 9126[10] and ISO 25010[11] software quality standards**

These standards contain criteria for evaluating the quality of software. The 25010 is the newer one that replaces 9126. They do not promote a testing procedure, but provide criteria that can be used for a checklist for software requirements and software tests/quality. Some of those criteria might be useful for the QA task.

## SCAPE project on QA

In SCAPE, a past EU project on Digital Preservation, the focus of the QA deliverables (D11.1, D11.2, D11.3) was on automated, scalable methods for quality assurance, in particular for digital media and documents. The tools developed in SCAPE focus on the aspects of media validation, comparison and repair in case of migration or bit rot; and policies for QA were defined with focus on validation.

The tools released by SCAPE[12] for QA are:

- Jpylyzer – JP2 validator and extractor
- Matchbox – Duplicate image detection tool
- xcorrSound – Improve your digital audio recordings
- Flint – Validate PDF/EPUB files against an institutional policy

The tools developed are certainly of interest, but have a different scope from our focus in PERICLES: as most tools developed in the context of digital preservation[13] are focused on content and digital object (DO) technical properties QA, in the scenario of migration or bit rot. Our focus is on the QA of the ecosystem model and the management of entities including policies and processes, as opposed to the building of processes for the QA of DO.

Some tools may be of interest to us for test cases where the QA of DO is also involved; Flint in particular could be of use for QA validation of PDF or EPUB documents according to a policy. QCTool[14], a tool for video preservation may be useful for the later task on high value digital media QA.

---

[10] Draft available at https://www.cse.unsw.edu.au/~cs3710/PMmaterials/Resources/9126-1%2520Standard.pdf

[11] Draft available at http://miageprojet2.unice.fr/@api/deki/files/2222/=ISO_25010.pdf

[12] http://www.scape-project.eu/tools

[13] http://coptr.digipres.org/Category:Quality_Assurance

[14] http://www.bavc.org/qctools

## 5.2.4.  Policies models, languages and standards

### High-level policies

High-level policies are usually specified as free text. There are many different forms of high-level policies which are often known as best practices, recommendation or just "guidelines".

These policies are targeted for a certain area. An exhaustive list of preservation policies[15] has been produced by the SCAPE project.

### SCAPE for Policies

The work done on policies in SCAPE has been detailed in deliverable D13.1 Bechhofer, S. et al (2013) and D13.2 Sierman B. et al (2014). The scope is intentionally limited to preservation policies, where the policies and systems are designed as preservation systems. In this task and PERICLES in general, we take a point of view where preservation is integrated into the existing systems and workflows; and not a separate system, as illustrated in our deliverable D5.1.

In SCAPE, policies are the input for the SCOUT tool, which is connected to a repository system and collects metrics. These metrics are stored on a dedicated store. The user-defined policies operate on top of that store. They check the events and collected data against policies and can send an alert in case of a policy violation.

This approach is different from the one in this task: SCAPE policies operate on controlled, collected data, in contrast with this task that operates at the model level. We aim to validate parts of the model as well as validating model instances. This is possible because the data elements of the ERMR (T5.1) refer to the models. The approach is generic, and as such it can be applied to different models. There are no constraints on how the model needs to be structured to use the QA methods.

In SCAPE, three levels of policy are defined by S Bechhofer et al (2013):

- **Guidance policy**: Very high-level statements which apply to the whole organisation;
- **Preservation Procedure policy**: Natural language human readable policy which may encompass the whole organisation or may be focused on a particular collection or material type depending on the needs of the particular organisation
- **Control level policy**: These are statements derived from the Preservation Level, which are in both a human readable and machine-readable form and relate to a specific collection or material type.

In our approach, we are not defining strict policy levels, but the base definition of policy can be easily extended to support a policy level hierarchy in general. We believe the levels will be use case specific, therefore we allow them to be added to a customised model without integrating them into the base ecosystem model.

---

[15] http://wiki.opf-labs.org/display/SP/Published+Preservation+Policies
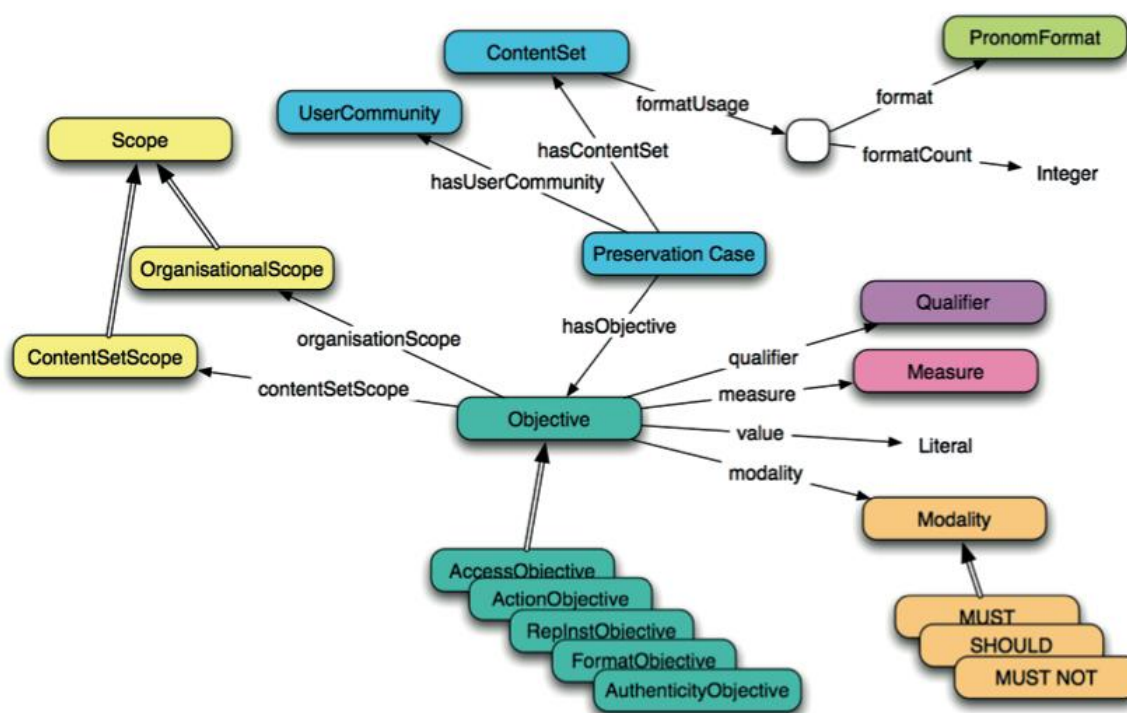
**Figure 2: Policy model in SCAPE from Kulovits et al (2013)**

As far as preservation policies are concerned, the SCAPE ontology (D13.1) is a good reference; for example, preservation objectives categories (access, format, authenticity… ) can be used to classify lower level policies.

The Catalogue of Preservation Policy Elements (D13.2 and online[16]) presents templates for a set of classes of guidance policies to preservation procedure policies in a defined template that includes control policies and examples of concrete policies. This is a valuable resource for building preservation policies and can be used as a reference: the classes of guidance policies can be used to classify policies in our policy model (paragraph 5.3 of this deliverable).

## Research Data Alliance (RDA) Practical Policy Working Group - Policy Templates, September 2014

The RDA Working Group has dedicated its attention to data policies and their implementation in data management systems. The template document from September 2014[17] defines policy templates aiming at computer actionable data policies. They include where possible, an exemplary implementation as GPFS[18] or iRODS[19] rule. The scoping of policies in the context of this group's outputs is specific to data management systems and aiming at low level, practical policies that include rule implementation.

---

[16] http://wiki.opf-labs.org/display/SP/Catalogue+of+Preservation+Policy+Elements
[17] https://b2share.eudat.eu/record/246
[18] IBM General Parallel File System: https://en.wikipedia.org/wiki/IBM_General_Parallel_File_System
[19] iRODS Integrated Rule-Oriented Data System: http://irods.org/

Each policy template contains:

- Policy name
- Example constraints that control application of the policy
- State information that is needed to evaluate the constraint
- Example operations that are performed by the policy
- State information that is needed to execute the operations

In this document the overall policy based data management model is also illustrated, and is specifically aimed at data management policies; with classes for replication, checksum, quota and data type policy. It is evident that this approach has a more precise scoping and is focused on the technical features and operations necessary for data management, and as such is very distinct from our task objective.

### SHAMAN

SHAMAN project deliverable D9.1 "Migrating the SHAMAN Preservation Environment", chapter 4 policies, deals with high-level (strategic) policies. The deliverable proposes to map policies to executable rules with a manual process. It suggests splitting big abstract policies into smaller components and associates them to the SHAMAN lifecycle model. The lifecycle consists of creating, assembly, archival, adoption and reuse. It is suggested to assign the sub-policies to the lifecycle phases.

### DIGITAL PRESERVATION POLICIES STUDY, Part 1

This study, Beagrie, et al., (2008) is written in the context of universities and colleges from UK and provides common preservation policies for that field. The study makes a suggestion on how to structure high-level preservation policies (p16f). The suggested categories are: principle statement (how the policy can serve the needs of the organisation's), contextual links (how it relates to other policies), preservation, objections (which preservation objectives are fulfilled), identification of content (to which content does the policy apply), procedural accountability (high-level responsibilities), guidance and implementation (how to implement the policy), glossary (definitions), and version control (history of the policy). There is a guideline and an example for each category.

## 5.2.5.    Rule languages

Rules are expressed in terms of IT mean machine-readable information. A rule can be a low-level representation of a policy and rule engines can provide the framework for running rules. There is a distinction between rules that originate from the security domain in form of access policies or as generic rules that can be used to change the behaviour of software. There are only a few vendor independent formal rule languages. There are different rule engines on the market which can be embedded into a software product[20]. Most of these rule engines have their own proprietary syntax.

Rules can be used for expressing preconditions and for the validation and formal language

---

[20] For example Drools (drools.org) and DTRules (dtrules.com) are open source rule engines.

specification of policy requirements, to express the implementation of low level policy statements, as already noted for example in Smith, M., Moore, R. W. (2007).

## RIF

The Rule Interchange Format (RIF)[21] mandated by the W3C is designed to be an exchange format between different rule engines. There are two RIF dialects (Kifer, 2008). One is Basic Logic Dialect (RIF-BLD) which uses horn clauses and the other is Production Rule Dialect (RIF-PRD) that uses production rules. The expression of RIF is XML.

## RuleML

The Rule Markup Language (RuleML)[22] that supports production rules with forward and backward chaining. RuleML is written in XML (Boley et al, 2001). The organisation contributes to SWRL and RIF standards. There are different experimental rule engines available.

## SWRL Semantic web rule language

SWRL[23] is designed for expressing rules for OWL ontologies (Horrocks et al, 2004). SWRL is not an official standard yet. It is still at a draft state by the W3C. The format is XML and is based on RuleML. Some OWL interpreters support SWRL, for example Protégé.

## ReAL

ReAL stands for "Resource Action Language"; it is the main topic of D3.4 Language of Change Management and is still an on-going work. Actions are logical combinations of RDF triple queries, insertion and deletion instructions and aim at updating the model, mainly in reaction of external changes of the ecosystem. Actions are triggered by events, and most importantly, can be combined within nested transactions in order to ease the specification of context-aware and globally consistent RDF modifications. The ReAL interpreter will be operated by an experimental LRM-oriented service (accessed through a REST API), and ReAL specifications will be part of the LRM model instance driving the behaviour of the service itself. The interpretation of ReAL actions will also rely on explicit inference mechanism when needed (querying functions). ReAL is designed to handle dynamicity in RDF store thanks to a much more adapted expressive power than standard alternatives based on production rules.

---

[21] http://www.w3.org/TR/rif-overview/
[22] http://www.ruleml.org
[23] http://www.w3.org/Submission/SWRL

# 5.3. Policy definition and model

## What we mean by policy

The word Policy is used in very diverse situations both in English, and in IT.

- A policy is a plan that defines the desired state inside an ecosystem. A policy describes the 'what' (guidelines) and not the 'how' (implementation). (PERICLES Glossary)
- A policy is a statement of intent, and is implemented as a procedure or protocol.[24]
- A formal statement of direction or guidance as to how an organization will carry out its mandate, functions or activities, motivated by determined interests or programs (Interpares[25])
- Many more specific uses, such as access control, security, load balancing, configuration policies

We adopt the generic PERICLES definition in a slightly more concrete form for task T5.3:

*A policy is a high and intermediate level natural language description of an institution's aims/goals that contain constraints on how to achieve the aims/goals.*

Policies can also be used to represent the legal requirements and aspects of an ecosystem.

We chose to use a generic definition that could fit more use cases, as opposed to a definition that would impose the use of a formal language or of a specific policy format or level. This is represented also in the Policy model here illustrated.

## Initial version of the policy model

The policy data model is defined independently of a specific ontology, but it has been implemented in the Digital Ecosystem Model through the process entity and related entities. A first version of this implementation is presented here in paragraph 7.3.4. We plan to refine the process entity model and present a final version in the scope of D5.3 Complete Tool Suite for Ecosystem Management and Appraisal Processes (M44).

The policy model will be used as the parent class for all types of policies, of all levels, and domain-specific sub-classes can be implemented by specialising this class. The entity is defined as follows:

- **Identifier**: a unique identifier for the policy
- **Name**: a friendly, not necessarily unique, informal name
- **Version:** version number (it can use the LRM versioning mechanism)
- **Description**: short description of the policy
- **Purpose**: the reason for creating the policy
- **Policy statement:** detailed definition of the policy contents as text (formal or not formal)
  - **Format**: formal; or non-formal (free text)
  - **Language**: the language used for the policy definition (natural, ReAL, SWRL, etc.)
- **QA criteria**: condition-action, rule, unit test or other formal definition of QA methods
  - **Format**: formal; or non-formal (free text)

---

24 https://en.wikipedia.org/wiki/Policy
25 http://www.interpares.org/ip2/ip2_term_fdisplay.cfm?tid=1021

      ○   **Language**: the language used for the policy definition (natural, ReAL, SWRL, etc.)

      ○   **Reference** to the QA criteria implementation as described in paragraph 5.5

      ○   **Trigger**: what will trigger the QA criteria validation (e.g. reference to the objects that will trigger evaluation on change)

- **Classification**: defines the type of policy; domain dependent, for preservation policies the SCAPE catalogue of policy elements [26] can be used as a reference.
- **Policy authority:** the owner of the policy, the entity that mandates the policy
- **Responsible** (**person**): responsible for the application of the policy
- **Sub-policies**: policies that are a more detailed specification of the parent policy as described in the policy derivation
- **Enforcers**: reference to the processes implementing and enforcing the policy
- **Level of compliance:** what is the desired level of compliance of the policy (must, should, must not); as defined in RFC 2119[27]
- **Current state of the policy**: how well the policy is currently implemented
- **Validity information:** any guidance to the policy lifecycle: Valid from; Valid to
- **Conflict detection attributes**: map of attributes for conflicting policies detection (see paragraph 5.7)
- **Target entities**: references to ecosystem and external entities affected by policy (depending on the policy level, consists of a free text description, a query, or a list of entities)
- **Target user community:** the user community the policy has been designed for
- **Replaced policy**: in case a new policy is created in order to replace an old one

# 5.4. Policy to process derivation

We propose to use policy derivation as a process to trace how the highest-level policies map to intermediate-level policies, down to concrete implementations such as rules, procedures, workflows and services with dependencies to other ecosystem entities. This is a method that will help rationalise the ecosystem structure by showing the dependencies between different level-policies, procedures/services and other ecosystem entities, and will enable policy-based methods for QA and validation of the ecosystem. This supports a managed change on all hierarchy level of the policies. We here illustrate some advantages of policy derivation:

- Enable evaluation of change impact
- Support policy conflict detection
- Can help define the ecosystem model from the top-down
- Support policy-driven systems

---

[26] http://wiki.opf-labs.org/display/SP/Catalogue+of+Preservation+Policy+Elements
[27] https://www.ietf.org/rfc/rfc2119.txt
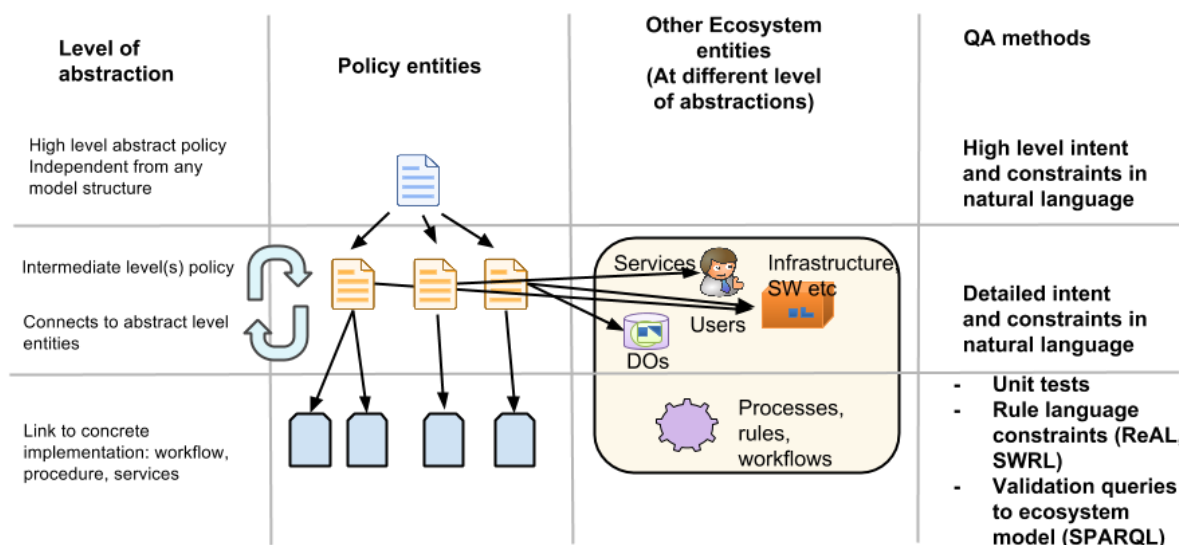
**Figure 3: a representation of the policy derivation model**

The policy-to-process derivation is a manual task. Interpreting the natural language and transforming it to executable policies automatically is only possible if the model imposes a controlled vocabulary. As a consequence, though, the expressivity of policies is severely limited. Here we prefer a more general model. This requires that a user maps the policy into executable forms in form of processes, and creates dependencies. One or more processes can implement a policy.

A short example of policy derivation is included in Appendix 4.

## Policy derivation guidelines

Given that policy derivation will be a manual process, we propose these simple guidelines to help consistent mapping of policies across levels.

Desirable attributes for policies:

- Clear purpose and focused
- As simple and clear as possible: few and explicitly listed exceptions, clear responsibilities
- Verifiable and measurable
- Make assumptions explicit

Steps:

1. Specify the affected digital objects or other ecosystem entities of the policy
2. Specify the target user community
3. Refine the policy into more detailed intermediate policies
4. Express the concrete implementation of the policy in natural language descriptions
5. Link the descriptions to the actual (code or rule or other form of) implementations
6. Create dependencies between the policy at all levels and the different ecosystem entities
7. Connect to events, actions, and policy evaluation triggers

# 5.5. QA criteria for policy and processes

In this paragraph we illustrate the different methods and criteria for QA we are proposing to support policy and model validation in the context of the ecosystem model. Use case scenarios for this task are available in Appendix V.

## QA of policies: Manual Dependency Checks (MDC)

QA for policies written in a natural language are very difficult to interpret and process automatically by the machine. For that reason a manual check has to be done by the Ecosystem Modeller to verify that a policy is completely implemented through its associated process entity, or the associated process entities of its sub-policies, if the policy is an aggregation of other policies.

> **Ecosystem Modeller [EM]:** The person in an institution who creates and updates the concrete Digital Ecosystem Model. There can be more than one person with this role. In this investigation we include the task of checking a policy for implementation completeness to the EM tasks, albeit this could also be done by a person who isn't modelling, but who is aware of the digital ecosystem ("Policy Manager").

> This perspective underlines the fact that a policy entity doesn't need to be related to a process entity directly, if it is completely implemented by the set of processes associated with its sub-policies.

> **Manual Dependency Check [MDC]:** We define the MDC as quality assurance method of one entity of the ecosystem model, which should verify that designated dependencies of the entity are fulfilled, and without contradiction. The MDC has to be executed manually by a human, usually the EM, who should be notified automatically when the check has to be done.

> **Audit Dependency [AD]:** An AD is a dependency between two or more ecosystem model entities, which need to be checked with a MDC, if one of the entities has changed.

The dependency between a policy and its implementing processes is an AD. If the policy entity or the associated process entity changes, then the MDC has to be executed to verify that all dependencies between the policy, sub-policies, and processes are still fulfilled. This can't be done automatically if natural language is used to define the policy.

## Advantage of MDCs

The integration of ADs into an ecosystem model means a higher workload for the EM because of the necessary introduction of MDCs. On the other hand it enables a half-automated ecosystem management which even includes the management of particular circumstances which can't be handled automatically, and which would need even more effort if handled completely manual.

A great benefit of this approach is that the moment in which a MDC has to be executed can mostly be detected automatically so that the EM can be notified directly at a critical moment. This can be done because the involved entities and their dependencies, excluding the ADs, are managed automatically. If there is a warning for a normal dependency of an entity, it can be resolved

automatically or manually…

1. …without affecting the entity, and therewith without triggering a MDC.
2. …with the effect of an entity change. In this case the MDC notification for all ADs of the changed entity is triggered.

This means for a policy entity that it has to be checked only if a child-policy, a process, or a child-process entity changes.

## Checklist for executing a MDC

MDCs can't be executed automatically, but the manual check can be simplified if a checklist is provided for the EM. For a process change triggered MDC, a checklist would look like this:

**The check is done, if:**

1. the policy which is associated with the process is still implemented by the process, or
2. the policy which is associated with the process is still implemented by the processes associated with all sub-policies

**Otherwise:**

Warning - The process has to be refined, or another process has to be introduced, so that the policy is implemented again.

Note:

- Parent-processes don't have to be checked, because this should work automatically. It could be that the parent-process is changed, too, but in this case another MDC would be triggered.
- Parent-policies don't have to be checked, because it is assumed that a process associated with a sub-policy only implements this sub-policy. The sub-policy is a defined part of the parent-policy, and once an associated and changed process is verified, this defined part should be covered again.

## Unit tests

We propose to use unit-testing principles from the field of software engineering to support QA of the PERICLES ecosystem model and its entities. Unit testing is a method of quality assurance for written source code during software development in which the unit tester writes tests for each source code unit, as methods and classes, to ensure their correct functionality. These tests comprise assertions about different states of the tested unit. A great advantage of unit tests is that they can ensure the correct functionality of single parts of source code, especially at the moment of source code changes. A similar proceeding is required as QA method for the ecosystem model to test the entities and to identify conflicts, dependency violations, and other problems especially in case of entity changes.

Unit tests are designed to run fast, to be executable separately for each single unit and to validate unit states using assertions. Furthermore unit tests help document the behaviour of a unit, because the developer has to describe in detail which behaviour is expected. This could also support the documentation of processes in the ecosystem model.

## Other methods that will be investigated in the task

We are proposing further methods for QA that will be investigated during the rest of this task, as briefly described in this list; we are not proposing an implementation of each of these methods but

to implement and experiment on the most promising ones.

1. Analysis of the ecosystem graph to determine weak spots in the ecosystem with respect to policy implementation (weighting based on dependencies and risk);
2. Creation of rule/action language constraints (ReAL, SWRL) to support policy QA and policy implementation: policy constraints can be expressed as rules or other form of constraints;
3. Validation queries to ecosystem model (SPARQL) to be run periodically - queries to the ecosystem model designed to validate the policy implementation and the structure of the model;
4. Methods to collect and evaluate logs from ecosystem components and test for errors and failures

# 5.6. Management of policy change

We define meta-policy as policy for managing policies, for example to express the behaviour to follow when changing policies. For example, a meta-policy could state, "revalidate the policy and its ecosystem dependencies upon change", or "run conflict detection when a new policy is introduced".

Although the LRM provides features and mechanisms to handle semantic versioning of policies and for tracking provenance of policy changes, there is the necessity to create a well-defined way of handling policy changes at the Digital Ecosystem Model, as illustrated in this paragraph.

## Type of change for policies

A first type of change is the one applied to the policy itself:

- Created: a new policy is introduced
- Modify/update (new version): can be for a number of reasons: changes in user requirements, legal requirements, change in standards, scheduled update, event, change in strategy of the owner, update to its QA requirements
- Inactivate: this can be done either with events or by schedules because a policy can have validity information attached.
  - Retired: event driven
  - Expired: scheduled
  - Superseded by: event driven
  - Replaced by newer policy
  - Deleted (if allowed in the domain of use)

A second type of change is a change on the processes implementing the policy, and another would be a change of entities which are constrained by the policy, or entities which are handled by processes implementing a policy.

## Handling of change

Policy entities are linked to process entities which ensure the application of the policy in the model, and which provide links to executable processes at the underlying ecosystem. We introduced three specialised process types at the model for the assurance of a well-defined change handling with due

regard to the policy entities. The execution of this process triple is triggered by one of the discussed policy related model changes. The process types are:

1. A process for model validation (**ProMV**): The purpose of this process is to get all entities from the model which are constrained by a policy.
2. A process for entity validation (**ProEV**): Each of these affected entities collected by ProMV are checked by ProEV regarding their validity of the policy. If one entity doesn't fulfil the policy requirements, then it is passed to the third process.
3. A process for entity transformation (**ProT**): This process takes an entity, which doesn't fulfil a policy, and transforms it into a valid entity producing a new version of this entity.

We are also discussing the introduction of more specific processes for the verification of other QA-criteria, e.g. temporal processes that execute "*cron job*" as periodically checks and more sophisticated methods.

## Example:

Consider that the following policy statement is introduced into the ecosystem model:

**PolicyIMG**:

"*All images on the organisation website must have a logo at the bottom right corner*".

The following figure shows the example of this policy together with the associated process triple, as discussed above. It depicts also the information flow between the processes.
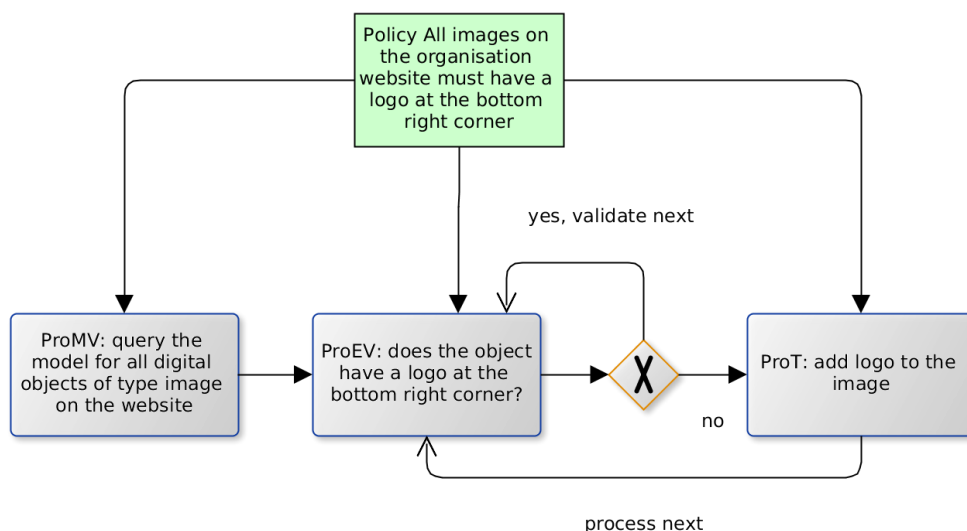


**Figure 4: Realisation of a policy with three distinct processes**

The realisation of a policy with three distinct processes has the benefit of flexible scheduling of the policy validation (ProMV and ProEV). It can be triggered if new entities that are constrained by the policy are added, either to the model or content that is referenced by the model. Or it can be scheduled, e.g. once per day.

The transformation process (ProT) can be used to enforce a policy in an automated way as it can

perform manipulation on the objects that failed the policy check. Of course it is also possible to notify other components or the user directly let the user make the choice on how to treat the failed entities.

### Change in a policy

A change in a policy can require change in the attached processes. The ecosystem is aware of the dependencies and can inform the user to verify if the processes still comply with the policy. It is assumed that there is a tool for this operation. One is for example the WP6 graphical Model Impact Change Explorer (MICE), which can visualize model changes. For example the policy statement of the explained example gets changed to:

**PolicyIMG_v2:**

> "*No images on the website must have a logo embedded*".

The user gets the request to change ProMV, ProEv and ProT. They do the following:

**Process for Model Validation (ProMV):**

> The validation process does not need a change. The user just approves this non-change.

**Process for Entity Validation (ProEV):**

> Here the process needs to be changed a checks of each digital object (image) according to the condition that no image must have a logo.

**Process for Transformation (ProT):**

> Also the transformation process needs a change. If an entity has a logo embedded, it fetches the master image and creates a new version (scaled down copy) that can be deployed on the organisation website.

After the user has changed the processes the three processes are executed, ideally in a sandbox with a prompt to the user. For this example it will present all images that have a logo embedded and ask the user if he wants to perform the transformation (remove the logo).

This approach ensures a compliant state of the model and the associated entities. The changed policies resides as active, it ensures the compliance not only during the change, but also in the future if new objects enter the system or due to other changes. A full, more detailed example of this change scenario will be provided on the upcoming D6.4 deliverable.

## 5.7. Policy conflict detection

A policy conflict arises if there are two or more policies that operate on the same set of entities and have contradictory criteria on what to do with the entities. Methodologies to detect conflicting policies vary depending on the form and implementation of policies.

It is common to have a conflict detection and resolution mechanisms in access policies. The evaluation of the policy conditions always leads to a definite access or deny result. One solution for avoiding conflicts is a ranking of access policies against a hierarchy. That means there are generic

policies in place, e.g. deny everything is the base policy. At a lower hierarchy level there is for example the policy "if the request originates from a specific network and port, then allow the request" defined. In this case the more specific policy has precedence. Other conflicts on policies of the same hierarchy level can be detected if there is an intersection between the targets and the resulting operation is different - access in one case and deny in the second case. There are also more sophisticated algorithms besides the hierarchy model for conflict resolution, such as in Huonder, F. et al (2010). In this task, we are using a different approach, as those methods do not apply to the generic policies. The result of a policy can be much broader than access and deny; and their expression is defined in a free form, ranging from natural language to domain specific languages.

For generic policies we define conflict as the situation where:

1. The set of target entities for a the policies overlaps;
2. The operation on the entities is conflicting.

Which operation could cause conflict on the policy statement? Because the policy can be expressed in natural language, it is often not possible to automatically analyse the statements.

It is useful to define a default set of conflicting operations/attributes, that can be manually or partially automatically (for lower level policies expressed in formal languages) picked and assigned to policies, and would then allow automated detection of conflicts.

## 5.7.1.    Policy conflict detection procedure

We propose the following procedure for determining when policy conflict arises:

A.        Manually define conflicting attributes and operations for a policy. For example:

| Attribute | Potentially conflicting attribute/operation | Target entities |
|---|---|---|
| keep content unmodified | operations that modify content: delete; add; modify content | Digital document preservation |
| keep bit stream unmodified | operations that modify bit stream: delete; add; modify; migrate format | Digital document preservation |
| preserve rendering environment | change rendering application | Digital document preservation |
| keep content available | maintenance work on technical infrastructure | Digital Objects, Technical Services |
| satisfy content access rights | content migration; change of access to technical services | Digital Objects, Technical Services, Processes |
| low response time of technical service | maintenance work; sophisticated calculations; high user workload | Technical Services, Communities |

B. Conflict detection

Given a set of policies; and the map defined in point A;

1. determine the sets of policies that have at least one conflicting attribute, based on the map from A (this could be implemented, in concrete terms, as a SPARQL query);
2. determine the set entities (target entities of the model) for each policy in each set;
3. for each set of entity from point 2, find the intersection;
   a. if the intersection is not empty, report policy conflict to the responsible for the policies
   b. otherwise, no conflict is detected.

## 5.7.2.    **Future work**

A further step can be that of assigning the attributes automatically (for example, some tests would allow determining if bitstreams, or contents of an entity are modified by a policy or operation).

Furthermore, it should be possible to identify possible conflict by looking at the input/outputs of processes, in case these overwrite some existing result; based on the current ecosystem model, it should be simple to implement as a query. At the model level we implemented a mechanism to express the level of compliance of a policy. A next step would be to add the possibility that the policy maintainer can mark policy conflicts based on the expected resolving, e.g. "This conflict won't be resolved, because it is not critical, therefore don't notify me again." Since there is mostly more than one method to implement a policy, conflicts often occur through the way a policy is implemented. Therefore an efficient conflict management has to analyse the implementing processes of a policy, too. For example, given a Policy A says to keep two copies of a Digital Object, and is implemented by a Process A, which makes two copies of it on the system. Policy B says that there shouldn't be more than one copy of a Digital Object on the system to spare disk space. The policies are not conflicting directly, because Policy A could have been implemented by keeping copies of the Digital Objects on an external server, but the current implementation of Policy A is conflicting with Policy B.

# 5.8. Implementation of the QA approaches (T5.3.2)

The practical implementation will work on the existing entity registry - model repository (ERMR). The models will describe the policy derivation, and the QA approaches will be realised by implementing the algorithms and methods described in 5.3.1. These will likely involve analysis of the dependency graph using different graph algorithms, or metrics, creation of unit tests, and execution of queries to the graph DB/triple store holding the ecosystem model, dependencies and entities.

Furthermore, changes at the underlying ecosystem have to be monitored to ensure that the model is in harmony with the system it represents. The PET tool[28] is one possibility for watching ecosystem changes, and we are investigating the mapping of its observations to the ecosystem model.

---

[28] Tool created in D4.1 of PERICLES for Environment Information Capture. Available at:
https://github.com/pericles-project/pet

# 5.9. Approaches to change management in semantics and user communities (T5.3.3)

Change is likely to affect cultural organisations in many ways, including internal and external factors. Building upon the explorations of semantic change within this project, we incorporate these insights into the development of change management approach.

Change management may include predictive and reactive elements: each is likely to be powered to a large extent by data-driven approaches. Hence, a prerequisite to effective management of change in semantics and user communities is the establishment and testing of methods capable of identifying on-going processes of change. Ideally, such methods approach - or even exceed - real-time detection speed (i.e. given methods with provable value as a predictive tool, it is possible to proactively respond to on-going processes of change).

In this task, we begin by evaluating PERICLES semantic drift detection tools against a number of data sets drawn from PERICLES use cases, in order to establish the effectiveness of these tools in real-world preservation contexts. We consider techniques for description of individual collections while preserving topology, such as the eSOM method described by Daranyi, Kontopoulos et al (in PERICLES WP4 D4.1); machine-learning techniques for comparative analysis of collections (see section 5.8 for further discussion); and techniques for forecasting collection composition into the future. In the first example (Figure 5), we can clearly see semantic change in terms of media artworks, resulting from greater freedom by the artist in the use of different media over time; we note the impact that this has upon classification terminology. The eSOM method explored here appears to be a very promising approach, which we are keen to pursue further in this context. In our second example (Figure 6), we study in more depth the behaviour of a particular medium, in this case 'acrylic', in order to examine acquisition behaviour over time of artworks categorised in this way, but also to (begin to) predict future behaviour.



| 1810-1900 | 1900-1980 | 1980-present |
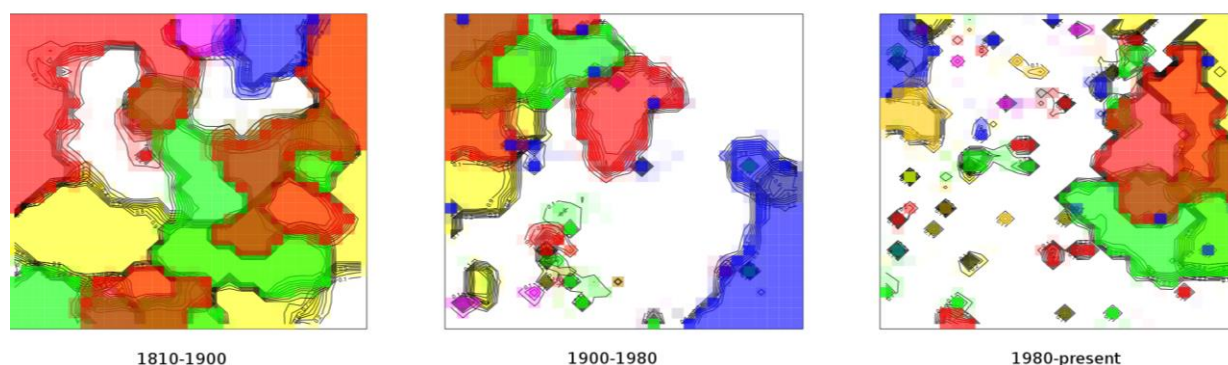
**Figure 5: Using self-organising maps to characterise the semantic space in a real-world image collection acquired between 1810-1900, 1900-1980, and 1980-present day. Three states of the evolving feature space of various media are shown here, including canvas (blue), ink (green), watercolour (red), wood (purple), graphite (yellow). Video artworks (gold) are visible only in the rightmost graph. As can be seen, traditional classification terminology for media artworks no longer suffice to describe the majority of items recently ingested into the collection; hence, classification termsets must necessarily broaden as a consequence.**

**Forecast of acrylic purchases**
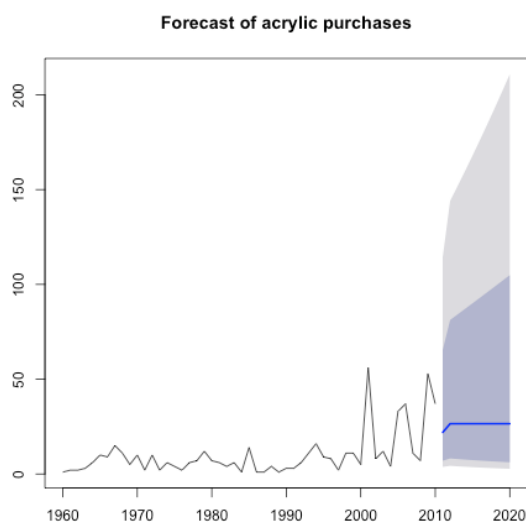


**Figure 6: Time series forecasting of purchases per medium; this time series forecast, implemented in R, applies an exponential smoothing state space model over a 50-year artwork acquisition dataset drawn from an online dataset (Github, 2015). This model does not consider seasonal variations, since acquisition data is generally reported yearly. Note that the uncertainty of the prediction increases rapidly over time, almost doubling within the ten-year period; hence, significant and increasing risk applies to forecasts over longer time periods. Forecasting can provide curators with useful intelligence regarding short-to-mid-trends within collection composition. However, where such predictions are used, the model uncertainty must be responsibly identified and communicated.**

Combined, these approaches are significant for change management in two main ways: First, they provide curators with information about past behaviours, which can provide useful insight into longer term trends, thus enabling curators to view their current situation within a larger context. This is particularly the case given the output of eSOMS analysis (e.g., of classifications); second, we can begin to provide curators with a glimpse into the future with the predictive forecast analysis. Although we acknowledge that this is crude, and that care needs to be taken in the interpretation of model uncertainty, we believe that such benchmarking at such a detailed level (e.g. predicted acquisitions for a particular medium) is valuable to curators. For example, it enables them to see how current behaviours will affect future activities, which will enable curators to assess whether this best meets future needs. As such it provides an empirical starting point for higher-level policy decision making. We also note that analysis of time-series to date can also show how previous environmental aspects have had an impact upon acquisition behaviour, and so enables curators to explore policy decisions alongside wider contextual factors.

# 5.10. Outlook

## 5.10.1. Task 5.3.1

This task has already analysed the state of the art for QA of policies, and change management, and refined the task definition and scoping from the DoW. We have further defined a policy model that fits the PERICLES ecosystem approach (already implemented in the ecosystem component), and we plan to release a final refined version by M44.

The policy to process derivation that has been described here will serve as a guideline for the creation of more concrete examples of policy derivation implementation in the ecosystem model. This will allow the creation of concrete examples including the policy and entity QA, and policy conflict detection using the methodologies outlined in the current deliverable, aligning with the overall PERICLES objective of validating the digital ecosystem reacting to change in its entities.

Policy change has been addressed also and feeds into a test scenario to be reported on in more detail in deliverable D6.4.

Finally, QA of high value Digital Objects will be investigated for digital artworks from the use case provider, by investigating how the artist's intent could be described in the form of policies, and how QA methods could be applied to the policy, for detecting issues with the policy implementation due to changes in the ecosystem or by conflicting policies.

## 5.10.2. Task 5.3.3

The methods currently being used for data-driven elicitation of policy require formal evaluation to establish the potential for usage of these methods in practical scenarios. Via an appropriate measure, it is useful to evaluate the accuracy and predictive power of forecasting methods and ensure that this can be communicated effectively to the users of the system. Similarly, the methods applied to characterise semantic spaces and for comparative analysis of collections require evaluation. They must fulfil the requirements of the task. It is also necessary to establish the parameters of use of such methods in a practical implementation. For example, a self-organising map in itself provides a visible indication of the level of fragmentation of a semantic space. However, in order to include such methods in PERICLES use cases, a dimensionally-reduced machine-accessible indication of this level of fragmentation may be desirable. As a second example, the strength of a visualisation may be evaluated via a user-cantered evaluation.

# 6. Support for appraisal processes (T5.4)

## 6.1. Objectives and definitions

### 6.1.1. Objectives

Appraisal is a process that in broad terms aims to determine which data should be kept by an organisation. This can include both decisions about accepting data for archival (e.g. acquisition) as well as determining whether existing archived data should be retained.

In traditional paper-based archival practice, appraisal is a largely manual process, which is performed by a skilled archivist or curator. Although archivists are often guided by organisational appraisal policies, such policies are mostly high-level and do not in themselves provide sufficiently detailed and rigorous criteria that can directly be translated into a machine executable form. Thus, much of the detailed decision-making rests with the knowledge and experience of the archivist.

With the increasing volumes of digital content in comparison to analogue, manual appraisal is becoming increasingly impractical. Thus there is a need for automation based on clearly defined appraisal criteria. At the same time, decisions about acquisition and retention are dependent on many complex factors. Hence our aim here is to identify opportunities for automation or semi-automation of specific criteria that can assist human appraisal.

To summarise the main objectives of the task are:

- To identify and define precisely a set of appraisal criteria whose evaluation is both relevant and can potentially be (partially or fully) automated.
- To provide methods and associated tools that automate the evaluation of specific appraisal criteria.
- To identify points in the content lifespan where appraisal (and reappraisal) is relevant and in particular, to demonstrate how appraisal is applied in changing environments.

In keeping with the overall PERICLES approach, the aim is to produce a focused set of tools running in a test-bed environment rather than a system.

### 6.1.2. Definitions and models for appraisal

#### 6.1.2.1. Types of appraisal

Within the context of the PERICLES case studies, appraisal can naturally be partitioned into two distinct categories.

- **Technical appraisal** – decisions based on the (on-going) feasibility of preserving the digital objects. This involves determining whether digital objects can be maintained in a reusable form and in particular takes into account obsolescence of software, formats and policies.
- **Content-based (or intellectual) appraisal** – acquisition and retention decisions or assignment of value based on the content of the digital objects themselves.

For many types of digital content, both types of criteria are evaluated. For example in the Tate Archive, a particular set of directories in an artist collection may be discarded because they are system files of no long-term value (content-based appraisal). A decision may be made not to acquire a software-based artwork, since there is a heavy dependency on custom software for which only the object code is available (technical appraisal). In the digital art field, technical appraisal is often referred to as *assessment*.

### 6.1.2.2. Continuum approach

In (Lagos et al, 2015), we defined a continuum approach to preservation, motivated in part by the Record Continuum theory in the related field of record keeping (Upward 1996). This comprises two main aspects.

- There is no distinction made between active life and end-of-active life; that is, preservation is fully integrated into the active life of the digital objects.
- Preservation is non-custodial, that is we do not aim to remove entities from their environment, both physical and organisational and place them in the custody of a third party.

This way of thinking about preservation in continually evolving environments reflects some aspects of the PERICLES case studies, which we describe briefly.

In the media case study, software-based artworks in a museum are in a continuous state of evolution due for instance due to hardware failure and software obsolescence. On one hand, a gallery such as Tate has a remit for long-term preservation of such artworks, whilst on the other they are often required for public display or for use by academic researchers. Potentially therefore multiple derivative versions of such artworks may exist, each of which requires appraisal.

In the science case study, space missions may run for several decades, from initial planning through to decommissioning. Much of the data created is required for active use by engineers and scientists as well as for longer-term preservation, and is under constant review as new observations are made.

The implications of this point of view is that we do not consider appraisal and selection only within a traditional lifecycle model (e.g. the DCC Lifecycle Model (Higgins, 2008)), when content items are appraised upon ingest to an archive. Rather appraisal and selection is regarded as an on-going process that can be triggered at multiple points during the existence of digital objects.

### 6.1.2.3. Policy-driven appraisal

The specification of appraisal criteria is performed through preservation policies, which we refer to as **policy-driven appraisal**. Since policies can be understood at different levels of detail from expressions of high-level organisational aspirations to low-level rules, we distinguish between different categories of policy, following an analogous pattern to the SCAPE project[29], which considered preservation policies in detail (see also section 5.2.4). The distinction here is that we are considering policies relating to acquiring and maintaining a specific set of content or "collection", rather than generic organisational policies as considered in SCAPE. Thus, policies are defined with

---

[29] http://www.scape-project.eu/

respect to a collection, a specific class of objects, generally defined through a specific genre or type, and also often reflecting organisational structures.

- **Collection policy** - a high-level aspirational policy, typically created at a higher level in an organisation that defines the overall goals and remit of a particular collection.
- **Collection strategy** - a more detailed and specific description of the objectives of a collection policy that includes information about how the objectives will be achieved.
- **Collection strategy implementation** - representation of a collection strategy in a form that is machine-processable.

The terms collection policy and collection strategy are widely used in the digital art community. In science, a collection may refer to a particular experiment or set of experiments.

## 6.1.2.4. Granularity

Appraisal can be considered at different levels of granularity such as collection-level, folder or sub-collection level and file-level. Manual appraisal at file level is often impractical due to the workload involved, whereas collection-level appraisal may result in content of no long-term value being retained.

In PERICLES, we are interested in model-driven approaches, and in particular notions of dependency between digital objects. Thus as far as possible, relevant information about collections is used to populate models, which can then be analysed for instance in performing the evaluation of appraisal criteria, without the need to work directly with the content itself.

## 6.1.2.5. Outputs and outcomes of appraisal processes

The outputs of the appraisal process vary according to the type of appraisal and content being considered. The output of content-based appraisal can be regarded as a statement of value or relative value. This may result in a specific outcome such as the deaccession of a particular set of content, but this may require additional human intervention. Technical appraisal can result in a statement of risk to objects in a collection, possibly with one or more implementable mitigating actions (e.g. replacing a piece of software or transcoding a video to a different format).

In the media case study, a gallery such as Tate has a limited capacity to acquire and manage digital content, so decisions need to be made based on the collection policies of the museum as well as available financial and staff resources for performing cataloguing.

On the other hand, in the science case study, all experimental data from the ISS is retained over the long term, even if they are marked as erroneous or of no value by scientists performing calibration experiments.

# 6.2. State of the art and appraisal criteria

In this sub-section we review the state of the art in appraisal, including potential definitions of appraisal and their relationship to the overall goals of the task.

## 6.2.1.     State of the art

Prior to commencing the technical work on appraisal, we conducted a comprehensive literature review. This is summarised briefly below.

### 6.2.1.1.     Definition and objectives of appraisal

The Interpares Project 2 provides a useful definition and structure for performing appraisal, which are quite similar to our viewpoint in PERICLES. It further provides checklist of factors that should be considered e.g. value, context, and authenticity. It defines appraisal as follows: *"to make appraisal decisions by compiling information about kept records and their context, assessing their value, and determining the feasibility of their preservation; and to monitor appraised records and appraisal decisions to identify any necessary changes to appraisal decisions over time."*

The (Paradigm, 2007) project lists important characteristics to consider in appraisal. These include the content of an archive, the context of the archive and whether the records have evidential value, the structure of an archive and the extent to which it sheds light on the business, professional or organisational prerogatives of the creator, and technical appraisal, i.e. can the institution maintain the digital records in a usable form. It also lists important cost factors.

The (DCC Digital Curation Manual Instalment on Appraisal and Selection, 2007) provides general guidelines on appraisal, based heavily on library practice. It defines "technical capacity to preserve" as a key appraisal factor. At the end of the manual are guidelines on how to develop an institution specific selection framework.

### 6.2.1.2.     Automation of appraisal

The University of Illinois (Metrics Based Reappraisal Project, 2014) proposes an iterative, technology-assisted, metric-based approach to appraisal. It takes into account usage statistics and other business performance measures for assigning a value score at the appraised resources, particularly aimed at automating appraisal of emails.

The Arcomem project (Risse & Peters, 2012) used the linked structure of web pages and the social web as a way of appraising and selecting content to be crawled. A further aspect of appraisal is to mine information about the trustworthiness and reputation of users from social web mining.

The PLANETS project developed tools and services for digital preservation (Farquhar, & Hockx-Yu, 2008), with a focus on experimental evaluation of preservation approaches within a controlled environment (Aitken et al, 2008). In particular, the PLANETS test-bed enabled evaluation of automated appraisal processes on large datasets, including an 'automated characterisation and validation framework for digital objects'.

Similarly, the SCAPE project, or Scalable Project Workflows, aimed to provide a framework for

automated, quality-assured preservation workflows (Edelstein et al, 2011).

## 6.2.1.3.  Risk management

Appraisal is often focused on characterisation of an object in its current state. However, a further dimension of appraisal is the effect of passing time: that is, the potential that events might occur in the future that limit the potential for preservation of material into the future. The DCC Digital Curation Manual identifies risk management as increasingly central to discussion of appraisal and selection (Harvey, 2006), permitting risks such as reduced accessibility, interpretability or ability to render material to be balanced against the consequences of that outcome. Traditional risk analysis is based on risk-impact (mitigation) analysis. This is a process, usually iterative, in which the following sequence of steps is typically taken: identification of risks; assessment of the severity and potential consequences of those risks (such as financial consequences, impact on schedule or technical performance, and so forth); planning for mitigation; implementation of mitigating actions based on the plan developed. As risks evolve, they are tracked.

The general-purpose project management methodology PRINCE2 specifies a series of steps in building and applying a risk management strategy (Bentley, 2010). A review of older methodologies for risk management may be found in (Raz and Michael, 2001). Risk management was brought into the forefront of preservation by the Cornell Library study into file format migration, reported by (Lawrence et al, 2000).

Many of the essential characteristics of a risk management toolkit were determined by PRISM (Kenney et al, 2002). Several existing risk management frameworks are explicitly intended to support preservation activities. These include DRAMBORA, the Digital Repository Audit Method Based on Risk Assessment (McHugh et al, 2008); TRAC (Trustworthy Repositories Audit & Certification: Criteria and Checklist, 2007), which includes risk-oriented terms in a checklist of key terms; TIMBUS, or Timeless Business Processes and Services (Vieira et al, 2014); and the SPOT (Simple Property-Oriented Threat) model (Vermaaten et al, 2012), which focuses on risks to essential properties of digital objects. Various tools are designed to support risk management in digital preservation planning, such as PLATO (Becker et al, 2008). A criticism that might be made of many of these tools is that the majority of such approaches do not focus explicitly on quantitative models, and rely on elicitation of craft knowledge in forecasting.

A considerable amount of recent research into risk analysis is available, much of which applies quantitative models in the forecasting of risk. Certain such models are appropriate for scenarios identified within PERICLES. Concretely, Stamatelatos (2000) recommends the use of probabilistic risk analysis for the deconstruction and evaluation of risk associated with elements of complex entities. For the analysis of events that have occurred to ascertain the cause, fault tree analysis may be used; for the analysis of events yet to occur, event tree analysis may be used. Zheng (2011) provides a detailed analysis of risk modelling in order to support decision-making in management of product obsolescence, which may straightforwardly be adapted to the purposes of forecasting and managing software obsolescence. Risk analysis may use publicly available resources for informational purposes; for example, Graf and Gordea (2013) demonstrate the use of DBPedia data to evaluate file format obsolescence.

(Falcao, 2010) provides a qualitative approach to risk analysis of software-based artworks, including a

number of detailed worked examples, which provides a valuable reference for such ecosystems.

### 6.2.1.4. Appraisal of art and media

A time-based media installation can be viewed as a dynamic system. The system transforms a media element into sounds and images, which are rendered to the viewer over time, within the context of a prescribed environment. (Laurenson, 2005) contains a question catalogue to determine the significance of a display equipment for a time-based media installation. Issues relevant to appraisal include identity and authenticity, and in particular what properties are essential for a particular installation to be a faithful instance of that work. In contrast to traditional art objects, where the aim is to minimise change to a unique physical object, elements of time media installations can often be changed (e.g. by the substitution of mass produced display equipment).

(Innocenti, 2012) discusses issues of authenticity in digital art, which is in a continuous state of evolution. This is relevant to appraisal since artworks will need to be continuously re-appraised as technology evolves. It describes the relevance of significant properties as capturing essential features of artworks that should be maintained.

Gathering information from the artist is an important step in guiding the preservation and appraisal of complex digital artworks. (Huys, 2011) discusses requirements gathering from artists for specific artworks.

The Rhizome ArtBase provides examples of policies and procedures that can be adapted into the traditional acquisition standards to specifically document the acquisition of software-based works. The (Rhizome Collection Management Policy, 1999) defines the organisation's mission, scope, acquisition, submission, acceptance, rejection, execution of ArtBase agreement and artist questionnaire, commission, removal of objects, removal procedures, distribution and copyrights, records, inventory and access to the collection. The (Media Art Notation System, 2013) (MANS) is a formal notation for describing media artworks. It is a specific flavour of MPEG-21 DIDL. It can be implemented at different levels from very high level to more granular. This is relevant to the technical appraisal of complex media objects such as software-based art as it describes the main elements and their relationships, including both digital and physical components. (Synchronised Multimedia Integration Language, 2008) (SMIL), a language that allows Web site creators to be able to define and synchronise multimedia elements, can also be applied to describe the behaviour of media artworks.

### 6.2.1.5. Appraisal of science data

Long-term appraisal of science data is an area that regularly attracts interest, particularly in the context of the establishment and maintenance of subject-specific data repositories such as the climate science repository run by the IPCC[30]. Similar examples include the ICPSR social science data repository[31] and the various subject-area-specific NASA data repositories, such as SEDAC (socioeconomic data), JPL (ocean circulation) and NSSDC (space science data centre). The approach

---

30 http://www.ipcc-data.org/docs/TGICA_DDC_Governance_2012feb08.pdf
31 http://www.icpsr.umich.edu/icpsrweb/content/datamanagement/lifecycle/selection.html

to appraisal taken, is reasonably specific and consistent within the subject-repository implementation pattern: in particular, guidelines often focus on the relevance, uniqueness, potential usability and use, level of documentation, level of accessibility, legal aspects and ease of replication of a dataset.

In some cases, geographic constraints are also identified as policies for acquisition purposes: for example, a national repository for social science data limits the provenance and coverage of data to data of relevance to that region. In others, funding source may be taken into account when appraising potential acquisitions.

Theme 8 Data Appraisal and Purge Prevention of the NASA (LTDP Earth Observation Guidelines, 2012) covers relevant standards and procedures for long-term data preservation of Earth observation space data. Only the CEOS data purge policy, USGS/EROS data appraisal process, ISO 14000 Environmental Management are deemed relevant for appraisal of such data.

## 6.2.2. Analysis of appraisal criteria

Broadly, appraisal criteria relate to the object under appraisal and to the environment in which the object is viewed. These appraisal criteria were derived from the material collected during the state-of-the-art survey described above; they were then subdivided into categories using the methodology previously successfully applied in the DELOS European FP7 project. The resulting table represents an extended adaptation of the conceptually similar analysis presented in the DELOS 4.3 Appraisal Report (Guercio et al). In this method, appraisal factors are categorised according to the data required for the factor in question to be evaluated: content, contextual, evidence, operational, societal, and technical.

The present analysis differs primarily in scope from that of the DELOS project; the latter focused on *appraisal* as *the determination of the worth of preserving information* (ibid.): that is, as a means of answering the question, 'what is worth keeping'? Here appraisal is considered as a process to be revisited throughout the life of the digital object. Consequently, our results differ principally in the breadth of material considered and in the number and breadth of appraisal factors identified. Whilst a large number of factors were identified during the literature search, only a small number are likely to be relevant to any given process. It is suggested that a prerequisite to the use of these factors in any specific scenario is the development of an application profile adapted to the scenario, including the subset of factors evaluated as relevant to that context.

For the purposes of brevity, we subdivide appraisal criteria into two key areas, as laid out in Section 6.1.2.1: technical appraisal, appraisal of the state of the object without regard to content-level relevance, and content-based appraisal, considering the object's intellectual relevance with respect to relevant extrinsic factors. In particular, selected examples of factors in content-based appraisal are included in the list below.

| Category | Appraisal factor | Example |
|---|---|---|
| Content | Significance | Significance with respect to policy, operational matters or functional matters |
| Content | Version | Determination of appropriateness of specific instance of object |
| Contextual | Intention of creator | Intentions of artist or creator with respect to the material |
| Contextual | Impact | Evaluated impact of object |
| Evidence | Precedence | Evidence of decisions made in the past which may be viewed as setting a relevant precedent |
| Operational | Policy | Fit with formal policy, as set/documented by relevant agencies |
| Societal | Historicity | Representativeness within class; relation to broader context |

**Table 1: Selected appraisal criteria, scored by project partners as of high importance (see Appendix 6 for a general listing of appraisal criteria)**

# 6.3. Appraisal scenarios

## 6.3.1. Methodology

In order to motivate the work on appraisal from an end user perspective, we describe four scenarios, two from each of the case studies that are used to provide exemplars and motivation for the technical development. These arose out of the initial requirements study in D2.3, as well as more detailed interviews conducted with staff from B.USOC and Tate as part of T5.4.

## 6.3.2. Media

### 6.3.2.1. Scenario M1

| Title | Risk assessment and mitigation for complex digital media art materials |
|---|---|
| Description | Technical appraisal of complex digital objects from the art collections or Tate Archive. |

| Description of content | The material for appraisal can include software and video-based artworks (SBAs respectively VBAs), as well as potentially complex digital objects occurring in the Tate Archive such as complex multimedia presentations and databases. SBAs and VBAs can also take into account risks to non-digital entities such as computer hardware and display devices, where these are relevant |
|---|---|
| Requirements for appraisal | The main goal is to maintain the long-term usability of digital objects in the face of obsolescence or failure. This includes estimating the risks of obsolescence or failure of the object, and determining impact, and evaluating potential mitigating actions. |
| Stakeholders and interaction | The main actor is a conservator who is responsible for maintaining artworks in a reusable form. |
| Timing | Such appraisal is performed on a periodic basis. This may include points at which a particular object is required for public access or exhibition. |
| Additional information | Additional input to the appraisal can be provided by the intent of the artist. This determines constraints or significant properties that may restrict the mitigating actions that can be performed, whilst maintaining the authenticity and integrity. |
| Outcomes | The main outcomes would be<br>● An assessment of risk to an artwork, and their impact on the reusability.<br>● Determination and cost estimation of potential mitigating actions. |

## 6.3.2.2. Scenario M2

| Title | Policy-driven content-based appraisal of artist collections (Tate Archive) |
|---|---|
| Description | Appraisal and selection of material from the Tate Archive. |
| Description of content | Content is sourced from the Tate Galleries data within Tate Archive, which is typically data that is acquired by Tate from artists' estates. Tate has no control of the structure or format of the data. These can include not only documents and multimedia content, but also more complex digital objects such as databases and multimedia presentations. Data can include relevant software as well as system files from the host machine. |

| | Typically datasets comprise thousands of files. Content may be provided on physical media of various types and, increasingly, data formats held on large storage volumes such as CDs or hard drives. |
|---|---|
| **Requirements for appraisal** | Appraisal of the Tate Galleries data is conducted pre-acquisition and post-acquisition. Typically an in-depth appraisal of the collections is only feasible post-acquisition. The pre-acquisition appraisal can be regarded as a cut down version of the post-acquisition stage.<br><br>Appraisal of Tate Galleries material is primarily driven by the relevant Collection Policy and Collection Strategy. |
| **Stakeholders and interaction** | The primary actor in this scenario is an archivist who makes acquisition or selection decisions. |
| **Timing** | Appraisal and selection of Tate Archive collections is typically performed pre- and post-acquisition. |
| **Outcomes** | The outcomes of the appraisal process can be<br><ul><li>Decisions about acquisition or retention of a particular dataset for a given collection.</li><li>Decisions about selection of a subset of a dataset for retention.</li></ul>Appraisal tools either provide an automated appraisal or selection decision, or provide information to assist an archivist in decision-making. In order to support the decision-making process, data visualisation tools may be used. |

## 6.3.3.   Science

### 6.3.3.1.   Scenario S1

| **Title** | Technical appraisal of stored science calibration experiments. |
|---|---|
| **Description** | Determine if an actual or potential technical risk to the data or its processing chain of a stored calibration experiment may require refactoring (i.e. reprocessing) of the data. |
| **Requirements for appraisal** | A science experiment comprises the input, and output datasets, potentially also intermediate datasets, the software used for processing the data, parameter files including details of the processing chain. The experiment can also include tools for |

| | evaluating the quality of the results. |
|---|---|
| **Detailed description of content** | A science experiment comprises the input, and output datasets, potentially also intermediate datasets, the software used for processing the data, parameter files including details of the processing chain. The experiment can also include tools for evaluating the quality of the results. |
| **Stakeholders and interaction** | Data managers, responsible for the management of collections of archived experimental data and software. Scientists expecting to reuse the data or pipeline in the future. Funders looking to ensure availability of the material in the future. |
| **Timing** | The process can be performed both at creation of the data as well as periodically throughout the lifetime of the content. |
| **Additional information** | This scenario is analogous to the technical appraisal of media content scenario M1. |
| **Outcomes** | <ul><li>Determine the risks to the re-execution of the experiment through pre-defined external factors.</li><li>Determine (and implement where appropriate) mitigating actions.</li></ul> |

## 6.3.3.2. Scenario S2

| **Title** | Hypothesis validation of science calibration experiments. |
|---|---|
| **Description** | Eliminate runs of an experiment by checking if they validate a given hypothesis. This might involve providing an algorithm to detect absurd measurements, and potentially what is wrong. In some cases, we may want to appraise intermediate results or do a backwards review of the data to identify the source of an error. |
| **Description of content** | The scenario is primarily aimed at experiments such as the SOLAR calibration experiments where a source dataset is processed using mathematical libraries, possibly using different algorithms and parameters, and the results of the various runs are compared and the quality is assessed. |
| **Requirements for appraisal** | In experimental contexts, it is common to include a validation step in runs of an experiment, in order to detect situations in which an experiment is returning inappropriate results. This can occur in situations in which the experimental setup is compromised: for |

| | example, if the lens of a solar sensor is occluded then the sensor does not receive sufficient light to measure solar intensity with accuracy. Identifying circumstances in which results are untrustworthy, and subsequently identifying the sources of error, is helpful in ensuring that invalid data is not published. |
|---|---|
| **Stakeholders and interaction** | Scientists performing experiments wishing to either validate their own results or those of other scientists.<br>Data curators who wish to ensure the integrity of data. |
| **Timing** | Performed throughout the lifetime of an experiment as results are created. |
| **Outcomes** | Provides a measure of the likely validity of both newly created data and completed experimental runs. |

# 6.4. Overall approach

## 6.4.1. Technical appraisal

In this sub-section we describe the approach to technical appraisal based on model-driven analysis, addressing scenarios. This is partly motivated by the definitions in section 5 of (PERICLES Deliverable D5.1, 2014), in which we defined in outline an approach to modelling digital ecosystems.

For the purposes of the discussion, a digital ecosystem can be viewed as a software-based artwork or a set of scientific experiments, together with their surrounding environment. An entity is a single element within that ecosystem such as a piece of software, a file or a display device.

We define two types of risks.

- A **primary risk** is a potential change to an entity arising through a stimulus that is external to the ecosystem.
- A **secondary (or higher-order) risk** is a risk to an entity as a result of a potential change to another entity on which it has a dependency.

We are primarily interested here in predictive rather than reactive approaches to modelling the impact of change. Projects such as PLANETS (Aitken et al, 2008) used a so-called *technology watch* to detect changes in the external environment, which could then result in changes to archived content. We are primarily interested in modelling risks through understanding longer-term trends to predict the impact of changes in the future. Thus our approach is primarily predictive rather than reactive, and is based on probabilistic models. As well as understanding the impact of potential changes, in some cases we are also able to determine mitigating actions.

Thus the main aims are

- to quantify primary risks to the ecosystem.
- to model the impact of primary risks on entities in the ecosystem.

- to model the impact resulting from higher-order risks through ecosystem models.
- to determine the mitigating actions with the least overall cost.

Using this approach, we aim to provide a tool for use e.g. by archivists, to analyse a digital ecosystem, determine at what point in the future there is a significant risk to the use or reuse of a digital ecosystem, to determine its potential cost impact and potential mitigating actions. Such a tool could be applied for example to assess the value of a software-based artwork, by determining how long it can be displayed in exhibitions before elements become obsolete or require refactoring, or the cost of maintaining a set of scientific experiments for a given time period.

Our overall approach to technical appraisal is summarised in the flowchart in Figure 7. Requirements for external information are shown in green arrow boxes.
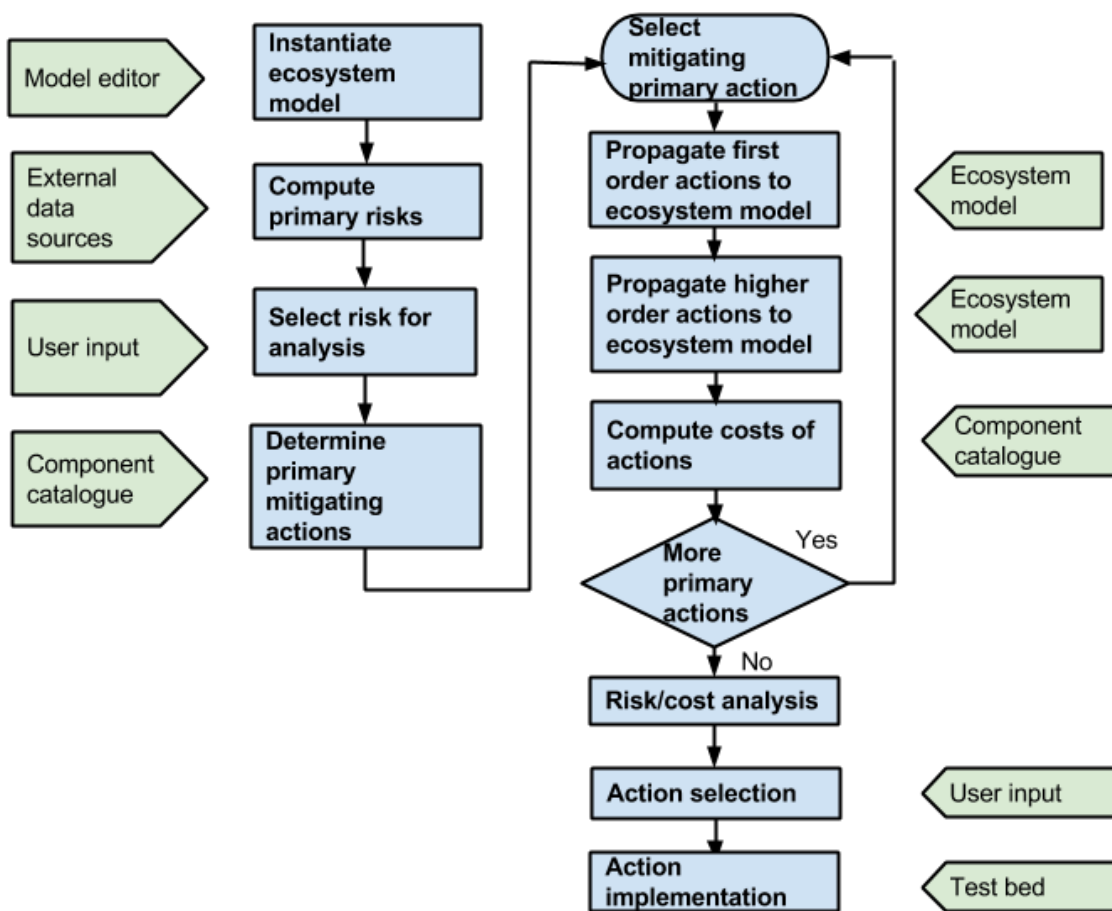


Figure 7: Technical appraisal workflow

The various steps will be described in more detail in the following subsections.

## 6.4.1.1. Entity categorisation

In order to understand and model the risks and mitigating actions to entities in a digital ecosystem, we produced a categorisation of ecosystem entities, a subset of which is described in Table 2. In order to model software-based and video-based artworks, after Falcao (2010), we also consider hardware entities. The entities cover the broad types hardware, software, data and user community

| Type | Abstract component | Description | Properties | Dependencies | Mitigation |
|---|---|---|---|---|---|
| Soft-ware | Operating system (COTS) | Commercial computer operating system | Manufacturer Version | Hardware requirements | Upgrade (major - new version, minor upgrade of current version) Virtualisation - same OS, but running on a VM Migration (to a completely different OS) |
| | Custom software application (executable only) | Inhouse software | OS requirements Hardware requirements Documentation | Operating system Hardware | Maintenance (e.g. by maintaining obsolete support software such as OS). Emulation (e.g. in different language) |
| | Custom software application (source available) | Inhouse software | OS requirements Source code language Documentation | Operating system Hardware | Upgrade (Modify existing software, using same language) Emulation (rewriting software to have same functionality in same or different programming language) |
| | Software application (COTS) | Commercial software application (assumed to be closed source) | Version Manufacturer Release schedule Support | Operating system Hardware | Upgrade (major, minor) Migration (to a different COTS application) Migration (to an existing open source application) Emulation (by custom software having same functionality) |
| | Software application (open source community) | Open source community software application (either commercial or free) | OS requirements Version Release schedule Source code language Documentation | Operating system Hardware | Upgrade (major, minor) Migration (to a different open source application) Migration (COTS application) Emulation (by custom software having same functionality) |

**Table 2: Classification and characterisation of PERICLES ecosystem entities**

Each of these component types may be characterised by appeal to a certain risk profile (Sandborn, 2007), which varies according to the manufacturing and sustainability model underlying the component. In large-scale software and hardware deployment projects, mature approaches exist for validating product sustainability prior to deployment (see Franch et al, 2013). Such approaches measure factors such as legal, regulatory and market aspects of the ecosystem within which a product exists, potential for migration between products and vendor commitment, where a vendor is present. Risk profiles are specific to entity types and a single curated object, such as a software-based artwork, may contain interacting components, to each of which a specific risk profile is attached.

A recent study found that component selection processes of this kind are not commonplace in science; software sustainability is not generally cited by scientists as a factor in the selection of software (Joppa et al, 2013), convenience, usability and existing profile in research publications being more likely to be referenced.

Strategies for the mitigation of software obsolescence (as opposed to failure, cf. Sandborn, 2007), not all of which will be relevant to a given software object, include purchase of source code, i.e. 'insourcing', redevelopment, re-hosting (migration to a new operating environment), maintenance and emulation.

### 6.4.1.2. Lifecycle of software and hardware

Various models exist to describe the lifecycle of software and hardware from manufacturing to obsolescence. Software models are typically based on the work of Halstead (1977), Putnam (1978) and Norden (1970), and make use of Norden's observation that data describing software development processes can typically be modelled using a certain mathematical distribution known as Rayleigh Curves. These distributions are similar to bell curves with an additional leftward skew, which reflects the observation that the majority of the effort spent in a software development process is spent in the earlier phases of the project, as illustrated in Figure 8. Pillai & Nair (1997) note that software development processes generally display a faster 'build-up' than hardware processes, so that accuracy is lost if the same model is used across domains.

To apply this model to real-life data, we make use of a mathematical generalisation of Rayleigh Curves, the Weibull distribution. This may be fitted to datasets taken from sources such as Google Trends or Sourceforge, using nonlinear fitting methods. Obsolescence models for reliability evaluation physical objects such as motherboards more often use 'bathtub functions' to estimate hazard (Klutke 2002).
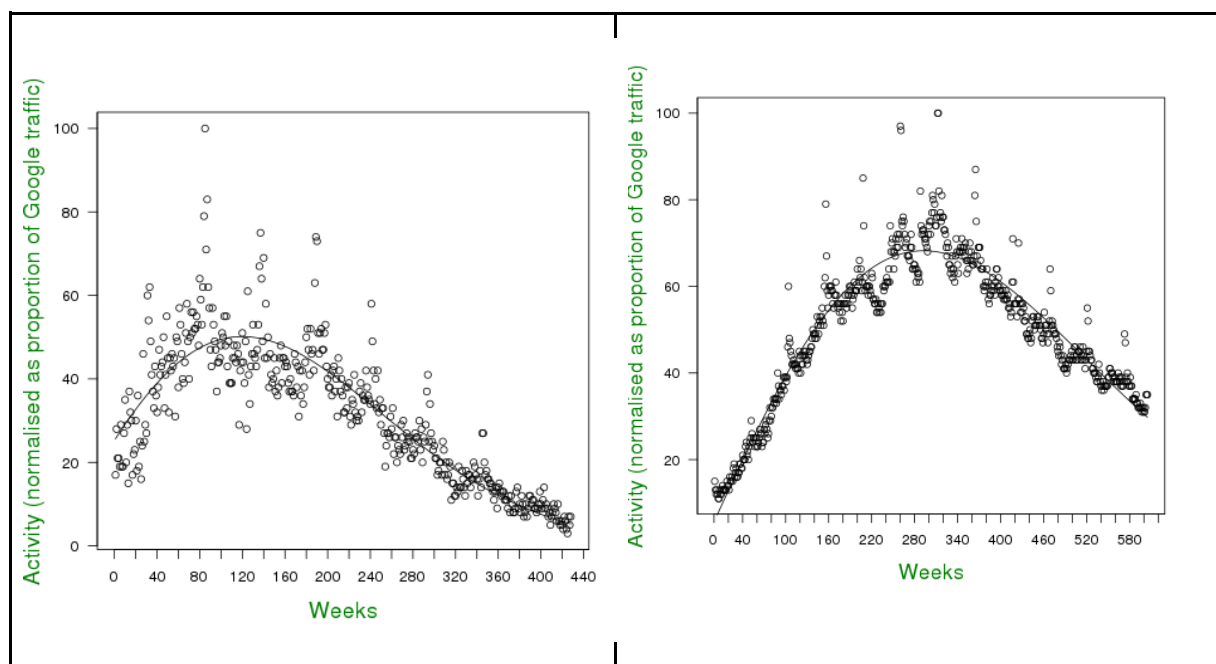


**Figure 8: Examples of Hazard functions**

Above left: Computation of hazard function for MediaTomb UPnP AV media server based on activity data extracted from Google Trends. Above right: Computation of hazard function for VLC Media Player based on activity data extracted from Google Trends.

As can be seen, MediaTomb is nearing the end of its lifespan; VLC, although attracting less attention than during its heyday, retains a significant time to live according to this model. Note that much of the 'noise' contained in this dataset is attributable to strong seasonal/periodic variation during each year. Using time series analysis it is possible to separate this variation from the primary data series.

The reliability of this model is currently under test: in this study, we track a number of COTS and open source applications and then calculate the statistical time-to-live of each. Comparing this to formally published end-of-line dates, where such exist, will permit us to evaluate the accuracy of this model following the completion of the study.

### 6.4.1.3. Entity design templates and rule-based dependencies

An analysis of a number of examples from art and science we conducted demonstrated that ecosystem models are built from a relatively small number of component types. Examples of such entity types are operating system, video file or display device. In order to facilitate models with interchangeable entity instances, we construct models using **entity templates**. These are ontology design patterns, which are essentially ontology fragments that can be used to model entities in a systematic way. This will greatly simplify the process of building ecosystem models. It also enables dependencies to be represented as rules between templates, which can then be evaluated against specific entity instances.

An example of an entity template was constructed in (Panagiotis et al, 2015), and is illustrated in Figure 9.
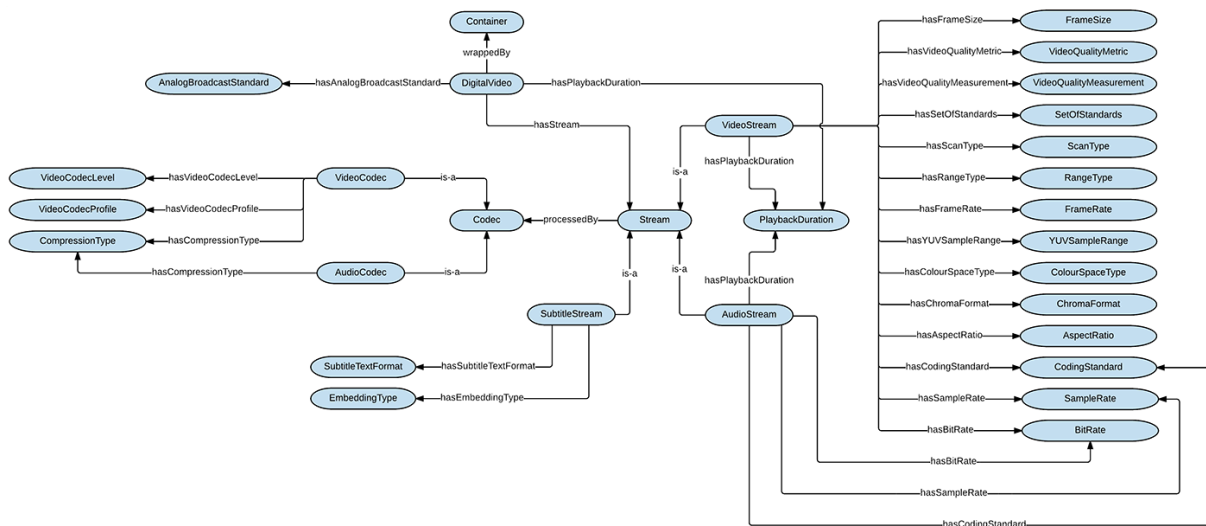


**Figure 9: Digital video entity template**

We now define the notion of dependency between two entity templates. This extends the definition of dependency produced in PERICLES Deliverable D3.2 Linked Resource Model. There are two types of property associated to an entity template.

- **Consumer properties** - properties that are exposed by outgoing dependencies. These represent the attributes of the entity that are required to be satisfied for the dependency to be fulfilled.
- **Supplier properties** - properties that are exposed to incoming dependencies. These properties or capabilities that the entity can offer to other entities.

For each supplier property there is a corresponding consumer property. Given two entity templates A and B, a dependency D of A on B is valid if a subset of the consumer properties of A satisfies a constraint (given by a rule) on the supplier properties of B. To motivate this definition, we consider two examples.

Video decoding is usually specified by constraints on formats, profiles and frame rates. For example, the inbuilt iPad Air video decoder has the following specification

- "H.264 video up to 1080p, 60 frames per second, High Profile in .m4v, .mp4, and .mov file formats;
- MPEG-4 video up to 2.5 Mbps, 640 by 480 pixels, 30 frames per second, Simple Profile .m4v, .mp4, and .mov file formats;
- Motion JPEG (M-JPEG) up to 35 Mbps, 1280 by 720 pixels, 30 frames per second, in .avi file format"

We then translate these conditions into rules (e.g. SWRL) that express a constraint between a video template and a video decoder template.

Such rules allow us to validate dependencies between abstract entities, so enable us to model hypothetical changes to entities in digital ecosystems such as replacement of a piece of software or transcoding of a video. Since dependencies between entities may be complex, such rules may only provide an approximation. However, they can enable us to make predictions, which can then be tested on the real entities themselves.

## 6.4.1.4.  Change propagation and impact analysis

Using the analysis of primary risks described in subsection 6.4.1.2, we aim in this piece of work to determine their overall impact on the digital ecosystem. This broadly follows the qualitative approach described by Falcao (2010), but following a quantitative method using statistical analysis and ecosystem modelling. Currently ecosystem models are developed by hand. Ultimately, the aim is to automate this process through the use of tools such as the PET, developed in WP4, and the PET2LRM mapping tool. The process of creating the models is simplified through the template approach described in subsection 6.4.1.3.

The initial aim of the risk analysis is to determine the secondary risks, which give an indication of the overall impact of primary risks on the ecosystem. The second stage of this work will incorporate analysis of recoverability or mitigating actions.

## 6.4.2.   **Content-based appraisal**

A variety of sources provide evidence for content-based appraisal activities. Policy documents are shared within organisations as guidelines for certain aspects of content acquisition. At a lesser granularity, broad aspects of collection policy are publicly shared.

For example, an archive may publicly focus on collections relevant to a particular broad theme or subject, such as '20th-century Scottish artists' or 'Charles Rennie Mackintosh'. An archive may work from a lengthy policy document (collection strategy) identifying individuals of specific interest within that broad mandate, which specifically guides certain appraisal decisions. Such policy documents are commercially sensitive and may remain confidential for that reason.

Existing material within a collection represents a key data source for content-based appraisal, since it enables characterisation of the substance of the present collection. That is, if a candidate item is under consideration for addition to a collection, several of the appraisal criteria (see for example the criteria extracted in Section 6.5.2) will reference characteristics of the existing collection. Does it replicate material already held? Alternatively, does it complement material held, or themes within the collection? Hence, knowledge about existing holdings is a prerequisite for some aspects of content-based appraisal; appropriate and rich characterisation of existing holdings is therefore a challenge of importance to content-based appraisal.

Furthermore, certain appraisal criteria require not only that material within current holdings is characterised, but also that material held by other institutions is considered: one such criterion is *Replaceability*, the ease with which an item can be replaced.

Our current work on the policy-driven appraisal scenario M2 is based on developing a number of sample collection strategies in collaboration with Tate, which can be used as a testing framework for machine processable implementations. Since current collection strategies are highly sensitive, the strategies produced for research purposes aim to reflect their essential features, whilst also enabling dissemination of results to the wider community. Interpretation of the collection strategies requires additional input from archivists Also, as mentioned above, additional background knowledge of the archivist may be used implicitly, and this can only be determined through close cooperation.

Content-based appraisal may be supported wholly or in part by automated processes. An example of the latter is supporting the archivist in rapidly characterising collections according to specific criteria. For this purpose, we return to the collection-level archiving scenario described in Scenario M2 (Section 6.3.2.2, above), in which large data collections are provided on a storage medium such as a DVD-ROM or hard drive to an archivist in order to allow for appraisal. This is a challenging problem for archivists, since the file system provided is ordinarily not well-organised; whilst the same is true of large paper collections, paper collections are essentially mono-dimensional, allowing the archivist to rapidly 'flip through' the material and gain a sense as to the shape of the collection. The hierarchical nature of a file system renders this difficult; furthermore, context and dependency of digital objects may not be immediately clear (Ross, 2012). Therefore, archivists may make use of tools intended for digital forensics in order to rapidly evaluate digital collections (see for example Kirschenbaum et al, 2010; Lee et al, 2012; John, 2012).
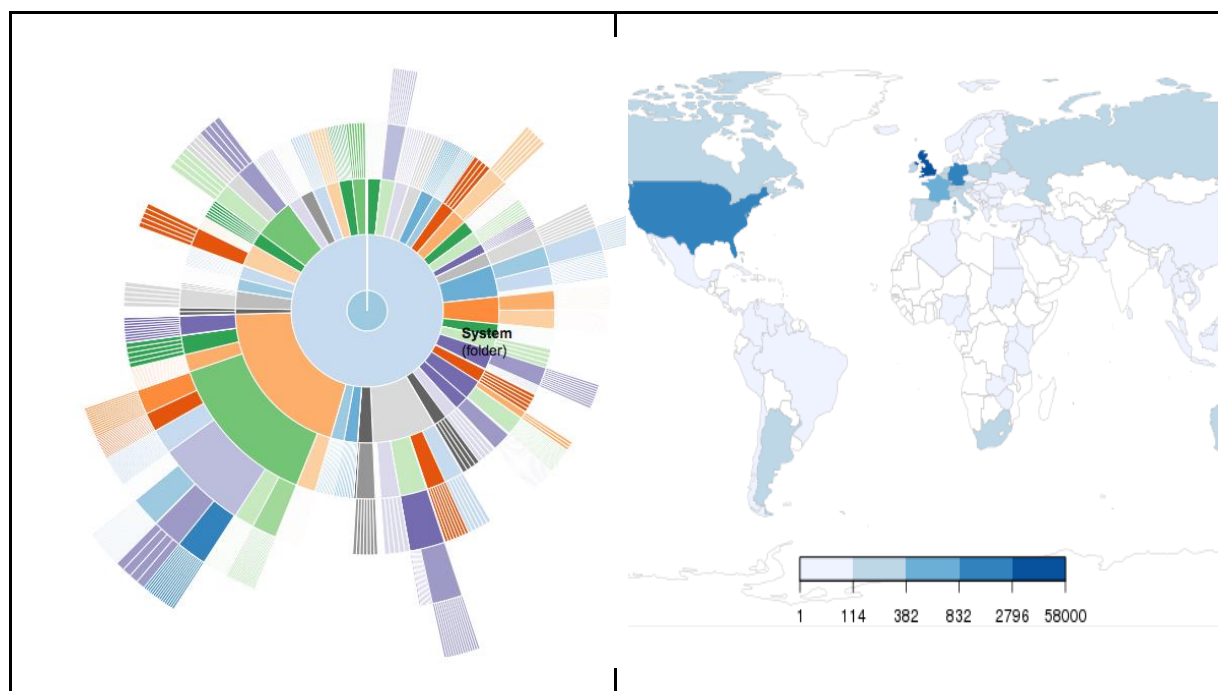
**Figure 10: Collection-level visualisations of material. Left: Metrics such as entropy may be used to support processes such as collection appraisal and review of complex hierarchies.**

The above prototype is implemented in the web framework D3. Right: Geographical metadata about material within a collection can reflect collection policy. This visualisation, implemented in R using geo-indexing based on resources taken from the Geonames project, displays the geographical origin of artists within a collection; in this case, it highlights prevalence of British and, to a lesser extent, US artists within the collection corpus (Github, 2015).

Wholly-automated content-based appraisal is likely to be of use in various scenarios in which 'preservation-focused maintenance actions' are indicated. It is particularly appropriate in scenarios in which periodic reappraisal of material on the basis of new evidence is felt to be indicated.

An example of such a scenario results from semantic change: as a result of such change, there is a need to periodically re-index material to better reflect the relevance of contemporary classification categories, enabling search and browse to function more effectively across a collection. Such an indexing process may be wholly automated.

# 6.5. Outlook for the task

We briefly summarise the outlook for T5.4.

## 6.5.1.    Technical appraisal

The next main step in the development of the technical appraisal is in developing techniques for the analysis of second order risks. This will draw on the quantitative analysis of primary risks and the ecosystem models. Initially this will be based on hand-crafted examples from the science and media case studies. We will also continue with the work on the entity categorisation and extend our initial

risk analysis to cover mitigating actions.

## 6.5.2.      Content-based appraisal

The focus on content-based appraisal will continue the development and analysis of sample collection strategies, in collaboration with Tate. We will explore and document the human interpretation of these policies with the archivists to produce a set of criteria that can be evaluated automatically, and produce appropriate technical prototypes. The aim here is to assist an archivist in making a decision about a collection using clearly defined criteria. A second aspect that can be explored further is the extent to which visualisation can be used to convey the results of appraisal. Initial concepts have been described above, but these require evaluations and testing with archivists to determine their effectiveness.

### 6.5.2.1.      Test bed implementation

In order to support the testing and integration of appraisal within the PERICLES test-bed, a simple test scenario has been developed in collaboration with UGOE in WP6. This is based on noise reduction within images, which can be used as a proxy for the space science calibration experiments conducted by the SOLAR scientists. It has the advantage of being simpler and uses open source software, which can more easily be modified. This scenario will be described in more detail in the deliverable D6.4 Final version of integration framework and API implementation due in month 33.

This initial development has enabled us to explore the capabilities of the test-bed and to ensure that the necessary hooks are in place to support the on-going integration of the appraisal tools during the final development phase of the project.

# 7. LRM based Digital Ecosystem Model (T3.5.2)

A first version of the ecosystem model has been presented in the D5.1 Initial report on preservation[32] ecosystem management deliverable. According to the new DoW this task has now been relocated to WP3 and the final version of the Digital Ecosystem Model (DEM) will be presented in D3.5 Full report on ecosystem management (M44).

This chapter briefly presents the current intermediate version of the DEM, as it provides also supporting functionality for the ecosystem tools. Furthermore a tool for the creation of DEMs will be presented in section 7.3.

## 7.1. Refinement of the ecosystem model

The goal of the ecosystem model is to model an existing system or procedure with the relevant entities, relations and dependencies in an abstract way. The model must contain all information to be able to trace the history of an existing system or procedure together with the activities that the users (communities) are performing. This proceeding enables a better documentation, preservation and reuse of activities, digital objects and infrastructure of the ecosystem, and supports an early detection and rectification of conflicts and failures. Furthermore this allows making reasoning about a change and the impact to the activities.

The Digital Ecosystem Model is fundamentally built on the Linked Resource Model. This is reasonable, because the DEM is a more application specific ontology compared to the LRM and can therefore reuse the generic modelling structure of the LRM.

| LRM base resource | DEM entity which inherits from the LRM resource |
|---|---|
| Resource | Ecosystem Entity |
| Agent | Ecosystem Agent |
| Action | Ecosystem Action |
| Event | Ecosystem Event |
| Dependency | Ecosystem Dependency |

Table 3: LRM base resources and ecosystem inherited resources

The above table shows an overview of the ecosystem entities and the LRM resources they inherit. The main advantage of the extensive use of the LRM for the DEM is that its modelling principles can easily be used at the DEM level, such as the LRM change management and semantic versioning. Furthermore this simplifies the later integration of the DEM into the test-bed architecture, because the PERICLES components are built with regard to the LRM. For instance the LRM Services will also be applicable to the DEM entities, because of the models use of LRM resources.

---

[32] Note that we changed the term to „digital ecosystem" as a result of the work done in D5.1 to embrace all digital ecosystems regardless of their status of preservation.

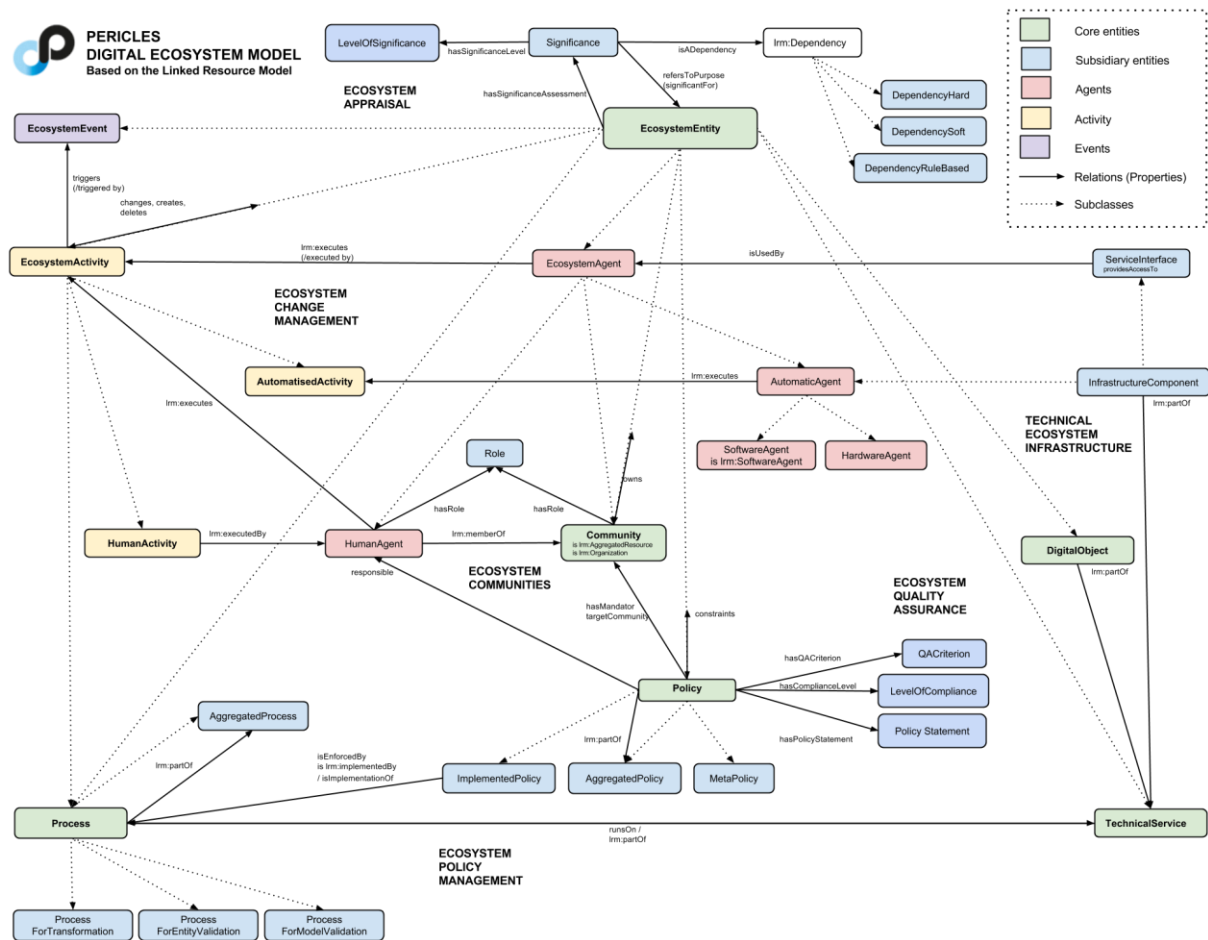The following diagram shows the current state of the ecosystem model:



**Figure 11: New version of the Digital Ecosystem Model**

## 7.1.1.     Main Entity Types

The first ecosystem model deliverable has introduced the five main entity types Digital Object, Policy, Process, Technical Service and User Community together with the theoretical concepts and definitions. These objects are quite generic and were now refined for a more detailed modelling. Below we present a short overview of the changes and new features.

### 7.1.1.1.     Digital Object

The Digital Object entity remains similar to the first version ("any item that is available digitally"). We highlight the possibility to define Digital Objects as Aggregated Resources, which is of importance for the new significance annotation mechanism (See paragraph 7.1.2 Dependencies and Significance).

### 7.1.1.2.     Policy

The Policy entity was discussed in detail in section 5.4 Policy to process derivation, therefore we give here only a short summary. A policy can now be described in more detail by a set of attributes which

support the quality assurance methods of task T5.3. For example it can be defined how exactly a policy is implemented at the underlying ecosystem, which QA criteria are of importance for the policy and which entities might be in conflict with the policy.

Furthermore it is now possible to determine responsible persons, policy authorities and target communities for a policy. There is an annotation for policies on the models to mark which policies are implemented by a process. Also meta-policies have been introduced, which enable the management of policies.
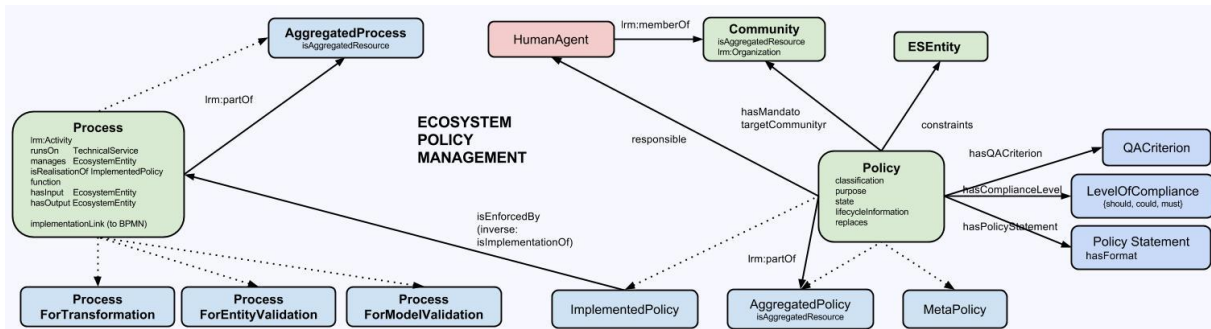


**Figure 12: Policy entity and the relation to other entities of the DEM**

### 7.1.1.3.    Process

Processes are executable activities of the ecosystem. The process flow is not modelled inside the DEM because there are already well suited notations for this task (e.g. BPMN). The process entity has been extended to have a link to such a process description. This enables automated process execution and combination of processes by the PERICLES testbed components within the underlying ecosystem in consequence of ecosystem model changes. A process can, but must not necessarily implement a policy.

Similar to a policy entity a process consists of aggregations of sub-processes, which enables the reusability of reliable ecosystem processes as parts of other processes. A process can now be modelled in more detail. This includes the purpose of the process and the input and output of a process. This is in particular necessary for the WP6 model compiler, which can combine sub-processes into a single process. Refer to paragraph 5.6 for a full description on how policies relate to processes.

### 7.1.1.4.    Technical Service

A technical service entity is level to the components (entities) it consists of digital objects and infrastructure components such as service interfaces and automatic agents, which execute automated activities on the ecosystem. These agents can now be expressed as hard- or software components in the DEM for a more detailed modelling.

Furthermore it can now be modelled, that service interfaces of a technical service provide well-defined access to the infrastructure components and digital objects, and that ecosystem agents and communities use them. The entities agents, activities and events of the DEM are derived from the Linked Resource Model and can be used together with other entities and relations from the LRM to express change in the model.

### 7.1.1.5. Community

The high-level entity user community has been renamed to community to encompass also groups of persons which are not the user community of digital objects, but also users that have other roles in the digital ecosystem (e.g. policy author, administrator). The original user community entity can be reintroduced to the model at any time simply by sub-classing the community entity.

By definition the community entity consists of one or more humans, which can be modelled explicitly with the human agent entity, if required. Communities and human agents can have certain roles in the ecosystem, which can be modelled with the role entity. A community usually owns other ecosystem entities and can be a policy authority, while the person responsible for a policy is a human agent.

In contrast to an automatic agent, a human agent is allowed to execute any ecosystem activities, especially also activities that change critical or important entities such as policies or communities.
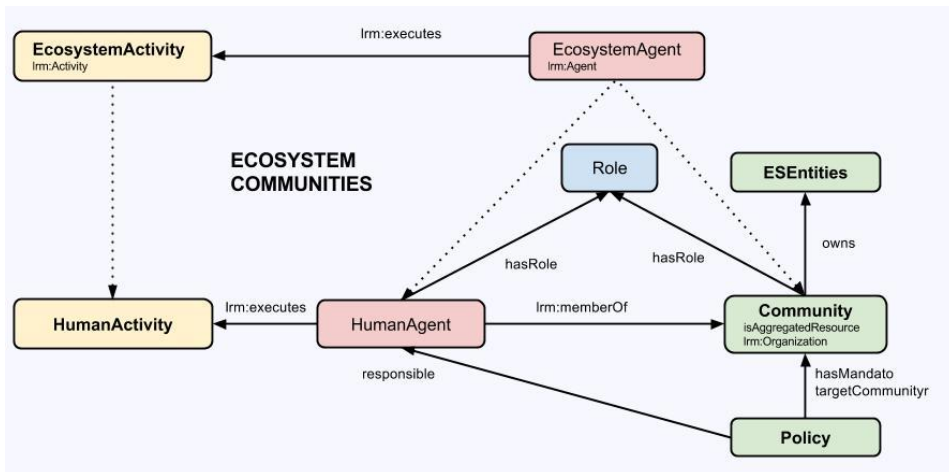


**Figure 13: Interaction of the entities "community" and "human agent"**

## 7.1.2. Dependencies and Significance

Another part of the model is the expressivity of the dependencies. Four main types have been presented in D5.1: hard, soft, fuzzy affinity, fuzzy range together with some additional properties. The latter two dependency types can be seen as a kind of rule that is attached to the dependency. Therefore they are unified as rule based dependency. Soft means a weak dependency and fits between hard and rule, it avoids having a rule attached.

The new version of the ecosystem model offers the possibility to annotate Ecosystem Entities as being significant, not significant, or dependent on other entities or for specific purposes. This is done with the significance entity, which can be linked to any lrm:Dependency, and therefore to all ecosystem dependencies, to express that the kind of significance behaves like a specific dependency. It is also possible to specify custom significance values, e.g. to express a customised level of significance of an entity, and to support graph based dependency calculations.

### 7.1.3.    Special properties of the entities

D5.1 Initial report on preservation[33] ecosystem management has presented several special properties of the base entities. This section explains briefly the current changes and addition. They are still work in progress and a full description with the theoretical backgrounds will follow in D3.5 Full report on digital ecosystem management (M44). D5.1 has added an annotation to dependencies to provide an estimate about when the dependency might change, the change probability and sensitivity. Change probability expresses how likely the entity can change and sensitivity shows how fragile an entity might be if the environment changes. Instead it has been replaced with significance because the values for sensitivity and change probability are hardly computable and mostly subjective to the person that creates the model. Also they are dependent on different time intervals.

We are currently investigating how to add the entities "risk" and "conflict" to save calculation values to the model graph. They will be designed similar to the significance entity, so that customised values, levels and types of these analysis entities can be specified. The algorithmic usage of the entities will be open to underlying algorithms, e.g. to implement the probability of change estimations as discussed in paragraph 5.3 on the model level.

## 7.2. Representation of the model

The deliverable D5.1 has presented the ecosystem model as a visual property graph. That means that both the vertices and edges can have multiple attributes. A representation in a formal language is required to work with the graph. There are three main options:

1. One is to use a generic graph framework that supports property graphs. It is a direct representation of a graph and the functionality is focused on graph operations. Querying and reasoning support is limited.
2. Another option is to directly record the information into a noSQL database that supports property graphs. This is not the case for all noSQL databases and the query language can vary between the different software vendors.
3. A vendor independent solution is to express the ecosystem model with the Resource Description Framework or as OWL ontology that is built on RDF.

The LRM uses OWL and provides functionality for capturing dependencies and trace the history of the models and other functionality like versioning and a set of default attributes for different predefined resource types. The features are also useful for the DEM and that is the reason why the ecosystem model derives from the LRM and extends its functionality. The LRM and the ecosystem model are abstract models, which need to be instantiated and adapted to model an intended use case.

---

[33]  Note that we changed the term to „digital ecosystem" as a result of the work done in D5.1 to embrace all digital ecosystems regardless of their status of preservation.

# 7.3. EcoBuilder: Digital Ecosystem Modelling Component

EcoBuilder is a PERICLES developed Java tool that uses the Apache Jena[34], an open source framework for handling RDF data. It provides a well-defined Java interface for the creation of scenario-based instantiations of the Digital Ecosystem base Model template. This procedure follows the mechanism of entity templates for the well-defined instantiation of the model and its entities, as described at section 6.4.1.3. The templates take care of LRM modelling conventions, so that a user who creates an ecosystem model instance can use this higher level of abstraction, without being aware of all details which makes the modelling simpler and less failure-prone compared to handwritten ontologies. The procedure with Java as additional abstraction layer allows the automatic generation of huge ecosystem models, for instance through the automatic parsing and mapping of database entries into ecosystem entities. The EcoBuilder can output the resulting RDFS/OWL based models and entities in different RDF formats, as Turtle or RDF/XML.

We also use the EcoBuilder to create the base DEM as it is a convenient way for us to follow an example-based test and refinement of the entities and principles of the DEM during the on-going development of the model. It helps us to bring the model into a mature state. In the following we will show an excerpt of the EcoBuilder guide, which provides basic examples on how to use the interface.

## 7.3.1. Creation of an Ecosystem Entity - A Digital Object of Interest

The EcoBuilder provides a Java template class for each possible resource of the DEM, and a data structure for its instances. The creation of ecosystem entities with the EcoBuilder can be done by creating Java objects from the corresponding resource class. These entity classes ensure the well-defined creation of the ontology resources and their instances. They specify the relations and properties that an entity can or must have, and enter the generated entity instances into the ecosystem ontology model.

Here we create an important file, which has the entity type of a digital object, and the version 1.0.

```java
DigitalObject importantFile = new DigitalObject(ecosystem,
    "ImportantFile");
importantFile.version("1.0");
```

**Figure 14: Creating an ecosystem entity**

This Java command will result in the following ontology resource:

---

[34] jena.apache.org

```
ecosystem:ImportantFile
        a               ecosystem:DigitalObject ;
        rdfs:label    "ImportantFile"@en ;
        lrm:version  [ a               lrm:Version ;
                       lrm:definition  "1.0"
                     ] .
```

**Figure 15: Generated ecosystem entity from previous figure (code)**

## 7.3.2.     Relations between entities - Modelling Technical Infrastructure

To model relations between entities the predefined Java methods of the entity classes are used. We will now create a server entity of the type technical service, and express that the important file lies on the server and can be accessed via a web interface.

```
TechnicalService server = new TechnicalService(ecosystem,
        "Server");
ServiceInterface webInterface = new ServiceInterface(ecosystem,
        "WebInterface");
server.hasPart(importantFile);
server.hasPart(webInterface);
webInterface.providesAccessTo(importantFile);
```

**Figure 16: Creating relations between ecosystem entities**

This code produces the following two new ontology resources "Server" and "WebInterface". You will notice that also the important file resource has changed - the lrm:partOf relation is used to express that the file is a component of the "Server" service. .

```
ecosystem:Server  a  ecosystem:TechnicalService ;
        rdfs:label    "Server"@en ;
        lrm:hasPart   ecosystem:WebInterface , ecosystem:ImportantFile .

ecosystem:WebInterface
        a                      ecosystem:ServiceInterface ;
        rdfs:label             "WebInterface"@en ;
        ecosystem:providesAccessTo  ecosystem:ImportantFile ;
        lrm:partOf             ecosystem:Server .

ecosystem:ImportantFile
        a               ecosystem:DigitalObject ;
        rdfs:label    "ImportantFile"@en ;
        lrm:partOf    ecosystem:Server ;
        lrm:version  [ a               lrm:Version ;
                       lrm:definition  "1.0"
                     ] .
```

**Figure 17: Generated relations - Output from previous figure (code)**

## 7.3.3.     Persons and Communities

Human agents and communities are fundamental parts of the ecosystem, as they have important roles in entity creation and ecosystem changes. We will create now a company entity as community, and the community of artists. Furthermore we will create a system Administrator which belongs to the company, and an artist, who is part of the artist's community.

```
Community company = new Community(ecosystem, "Company");
Community artists = new Community(ecosystem, "Artists");
company.owns(server);
HumanAgent admin = new HumanAgent(ecosystem, "SystemAdministrator");
HumanAgent artist = new HumanAgent(ecosystem, "Artist");
artist.isMemberOf(artists);
admin.isMemberOf(company);
Role dataCreator = new Role(ecosystem, "DataCreators");
dataCreator.describedBy("Persons who create Digital Objects");
artist.hasRole(dataCreator);
importantFile.hasAuthor(artist);
```

**Figure 18: Creating communities, human agents and roles for them**

There are now five new entities in the ecosystem model: The two communities, two human agents
and a role entity. Also the Important File entity changes again and gets a relation, which points to the
artist entity as author of the file. Since the artist created the important file we assigned the role
entity data creators to the artist entity.

```
ecosystem:Company  a    ecosystem:Community ;
        rdfs:label      "Company"@en ;
        ecosystem:owns  ecosystem:Server ;
        lrm:hasMember   ecosystem:SystemAdministrator .

ecosystem:Artists  a    ecosystem:Community ;
        rdfs:label      "Artists"@en ;
        lrm:hasMember   ecosystem:Artist .

ecosystem:Artist  a       ecosystem:HumanAgent ;
        rdfs:label        "Artist"@en ;
        ecosystem:hasRole  ecosystem:DataCreators ;
        lrm:isMemberOf     ecosystem:Artists .

ecosystem:SystemAdministrator
        a               ecosystem:HumanAgent ;
        rdfs:label      "SystemAdministrator"@en ;
        lrm:isMemberOf  ecosystem:Company .

ecosystem:DataCreators
        a               ecosystem:Role ;
        rdfs:label      "DataCreators"@en ;
        lrm:specification  [ a            lrm:Description ;
                             lrm:definition  "Persons who create Digital Objects"
                           ] .

ecosystem:ImportantFile
        a                 ecosystem:DigitalObject ;
        rdfs:label        "ImportantFile"@en ;
        ecosystem:hasAuthor  ecosystem:Artist ;
        lrm:partOf        ecosystem:Server ;
        lrm:version       [ a            lrm:Version ;
                            lrm:definition  "1.0"
                          ] .
```

**Figure 19: Generated community and human agent entities - Output from previous figure (code)**

## 7.3.4.     Policy, Processes and Quality Assurance

Policies are the most complex ecosystem entities, as shown in chapter 5.

```java
ImplementedPolicy policy = new ImplementedPolicy(ecosystem,
        "AccessibilityPolicy");
policy.describedBy("All important files should be "
        + "accessible via the web interface.");
QACriterion accessibility = new QACriterion(ecosystem,
        "ServerAccessible");
QACriterion availability = new QACriterion(ecosystem,
        "FileAvailable");
policy.hasQACriterion(accessibility);
policy.hasQACriterion(availability);
policy.hasMandator(company);
policy.responsiblePerson(admin);
policy.hasComplianceLevel(ComplianceLevel.MUST);
policy.setState("implemented");
policy.constraints(importantFile);
policy.constraints(webInterface);
policy.constraints(server);
```

**Figure 20: A Policy entity with quality assurance criteria**

In this example we create a policy, which ensures that the important file is accessible via the web interface. Furthermore, we define the company as policy authority and the system administrator as responsible human agent. The policy is linked to two quality assurance criteria: One defines that the server should be accessible and the other says that the file should be available.

```
ecosystem:AccessibilityPolicy
        a                       ecosystem:ImplementedPolicy ;
        rdfs:label              "AccessibilityPolicy"@en ;
        ecosystem:constraints   ecosystem:Server , ecosystem:WebInterface ,
        ecosystem:ImportantFile ;
        ecosystem:hasComplianceLevel [ a              ecosystem:LevelOfCompliance ;
                                ecosystem:complianceLevel  "must"
                                ] ;
        ecosystem:hasMandator   ecosystem:Company ;
        ecosystem:hasQACriterion ecosystem:FileAvailable , ecosystem:ServerAccessible ;
        ecosystem:hasState      "implemented" ;
        ecosystem:responsible   ecosystem:SystemAdministrator ;
        lrm:specification       [ a            lrm:Description ;
                                lrm:definition  "All important files should be accessible
                                via the web interface."
                                ] .

ecosystem:ServerAccessible
        a           ecosystem:QACriterion ;
        rdfs:label  "ServerAccessible"@en .

ecosystem:FileAvailable
        a           ecosystem:QACriterion ;
        rdfs:label  "FileAvailable"@en .
```

**Figure 21: The generated policy with two QA criteria - output from previous figure (code)**

The next steps for the EcoBuilder are

- Adaption to the final LRM version, once ready
- Integration into the PERICLES test bed architecture
- Release as open source tool

- [Out of scope for PERICLES] Implementation of a graphical interface for modelling

# 7.4. Conclusion and outlook

There will be a final version of the ecosystem model in deliverable D3.5 Full report on digital ecosystem management (M44). A full explanation of the entities, properties and changes since D5.1 will be provided. An on-going task is the further refinement of entities for the usage in other WP3/5 tasks. This is for example, relevant for the structure of T5.3 QA policies and for the attributes of the process object, which are used by the T6.2 process compiler. The deliverable will also provide more examples with test cases that show the reasoning applied to the ecosystem model. Also the ontology model will use more LRM concepts as soon as the LRM language concepts evolve, e.g. semantic versioning, the use of the ReAL rule language and tracing activities with LRM. In addition to this, a set of events and triggers will be considered. They can be used if a software component should drive the model.

# 8.   Digital Ecosystem tools conclusion and outlook

This deliverable has presented the current state of the individual tasks, which are part of the digital ecosystem tools. Next follows the final Deliverable D5.3 Complete tool suite for ecosystem management and appraisal processes describing the digital ecosystem tools which will cover the finished approaches of the different areas. Subject to the topics, the work will continue in the following way:

The PERICLES Entity Registry (T5.1) is a central component that has started as a file store and has now been extended to support registration and models. The next parts are integration with the test scenarios and test-bed, the trigger and notification system to support model evaluation and extension of the query interface for the model needs. The communication with the upcoming LRM service is also a future topic. Process support and execution that follow LTDP best practises is also a major task (T5.2).

The quality assurance task (T5.3) has introduced how QA approaches can be added to models. The method we presented is independent of a specific programming model or policy language. We have developed an initial version of a QA policy model that includes support for conflict detection and introduced a way how policies can be mapped to processes. Some demonstration examples will follow. Another topic is change management. One half-automated and one manual method have been introduced. The work will continue and some produced test cases will demonstrate the methods. Parts of the work from this task will be included in the Digital Ecosystem model. Also an initial view of semantic change (management) has been presented. Cooperation with appraisal task will follow on risk management for QA.

The appraisal task (T5.4) has investigated on how manual appraisal works, which policies to apply for appraisal and to which categories the policies belong. Appraisal can be divided into technical appraisal and content-based appraisal. Technical appraisal will be based on a model driven approach, while content-based appraisal will be based on the categorisation of the content. For technical appraisal work will continue on identification of changes on entities that may affect other entities (second order risk). The aim for content-based appraisal is to produce criteria for automatized evaluation and also visualisation. There will be a close link with the case studies.

Some appraisal scenarios will be transformed into an executable test case for the WP6 test-bed, for example noise reduction within images.

A full description of the Digital Ecosystem model (T3.5.2) will follow in the D3.5 deliverable. In the meantime the entities will be refined and more concepts from the T5.3 QA will be added. Also certain concepts from the WP2 domain ontologies may have relevance for the model. By this time the LRM model and ReAL rule language will be finished, so further additions will follow. Another part is the definition of events and action. Examples and guidelines for the DEM will also be provided.

Also part of the next deliverable will be the policy editor (T5.1.2), which allows expressing policies

with a controlled vocabulary. This component will also be integrated into the test-bed.

# 9. Bibliography

Aitken, B., Helwig, P., Jackson, A., Lindley, A., Nicchiarelli, E., Ross, S. (2008). "The Planets Testbed: Science for Digital Preservation" , The Code4Lib Journal, Issue 3, 2008-06-23. Web. Accessed 3rd Sept., 2015. *http://www.dcc.ac.uk/resources/briefing-papers/technology-watch-papers/planets-testbed#sthash.mRsLQkcg.dpu.f*

Beagrie, N., Semple N., Williams P., Wright R. (2008). DIGITAL PRESERVATION POLICIES STUDY, Part 1: Final Report October 2008. Web. Accessed 20th August, 2015. *http://www.webarchive.org.uk/wayback/archive/20140615022334/http://www.jisc.ac.uk/media/documents/programmes/preservation/jiscpolicy_p1finalreport.pdf.*

Bechhofer, S. et al (2013) "Final version of Policy specification model", SCAPE Project deliverable, *http://www.scape-project.eu/wp-content/uploads/2013/08/SCAPE_D13.1_UNIMAN_V1.0.pdf*

Becker, C., Kulovits, H., Rauber, A., & Hofman, H. (2008, June). Plato: A service oriented decision support system for preservation planning. In *Proceedings of the 8th ACM/IEEE-CS joint conference on Digital libraries* (pp. 367-370). ACM.

Bentley, C. (2010). *Prince2: a practical handbook*. Routledge.

Boley, H., Tabet, S., & Wagner, G. (2001, July). Design Rationale for RuleML: A Markup Language for Semantic Web Rules. In *SWWS* (Vol. 1, pp. 381-401).

DCC Digital Curation Manual Instalment on Appraisal and Selection (2007). Web. Accessed 4th August, 2015. *http://www.dcc.ac.uk/sites/default/files/documents/resource/curation-manual/chapters/appraisal-and-selection/appraisal-and-selection.pdf.*

Edelstein, O., Factor, M., King, R., Risse, T., Salant, E., & Taylor, P. (2011). Evolving domains, problems and solutions for long term digital preservation. *Proc. of iPRES 2011*.

European Commission (2012) Proposal for the EU General Data Protection Regulation.

Falcao, P. (2010). *Developing a Risk Assessment Tool for the Conservation of Software-based Artworks* (Doctoral dissertation, HKB).

Farquhar, A., & Hockx-Yu, H. (2008). Planets: Integrated services for digital preservation. *International Journal of Digital Curation*, *2*(2), 88-99.

Franch, X., Susi, A., Annosi, M. C., Ayala, C. P., Glott, R., Gross, D. & Siena, A. (2013). Managing Risk in Open Source Software Adoption. In *ICSOFT* (pp. 258-264).

Github (2015). Collection Metadata. Retrieved 2015-01-01 from https://github.com/tategallery

Graf, R., & Gordea, S. (2013, September). A risk analysis of file formats for preservation planning. In *Proceedings of the 10th International Conference on Preservation of Digital Objects (iPres2013)* (pp. 177-186).

Great Britain (1998) Data Protection Act. London: Stationery Office.

Guercio, M., Oliver, G., Pala, C. and Ross, S. (2008). Delos Appraisal Report 4.3.

Halstead, M. H. (1977). Elements of Software Science. Amsterdam: Elsevier North-Holland, Inc. ISBN 0-444-00205-7.

Harvey, R. (2006). DCC Digital Curation Manual: Instalment on Appraisal and Selection.

Higgins, S. (2008). The DCC curation lifecycle model. *International Journal of Digital Curation*, *3*(1), 134-140.

Horrocks, I., Patel-Schneider, P. F., Boley, H., Tabet, S., Grosof, B., & Dean, M. (2004). SWRL: A semantic web rule language combining OWL and RuleML. *W3C Member submission*, *21*, 79.

Huonder, F., Joller, J. M., & Rüschlikon, Z. H. (2010). Conflict detection and resolution of XACML policies. *Master's thesis, University of Applied Sciences Rapperswil*.

Huys F. (2011). The Artist Is Involved! Documenting Complex Works of Art in Cooperation with the Artist, P105-118. In Inside Installations. Theory and Practice in the Care of Complex Artworks edited by Tatja Scholte and Glenn Wharton. Amsterdam University Press, 2011. *http://www.oapen.org/download?type=document&docid=467012*.

Interpares Project 2 (2007). Chain of Preservation (COP) Model, Deliverable A4.2. Web. Accessed 4th August, 2015. *http://www.interpares.org/ip2/ip2_model_display.cfm?model=cop*.

ICO, 2015. Sending personal data outside the European Economic Area (Principle 8). *https://ico.org.uk/for-organisations/guide-to-data-protection/principle-8-international/*.

Innocenti, P., Ross, S., Maceviciute, E., Wilson, T., Ludwig, J., & Pempe, W. (2009, October). Assessing digital preservation frameworks: the approach of the SHAMAN project. In *Proceedings of the International Conference on Management of Emergent Digital EcoSystems* (p. 61). ACM.

Innocenti, P. (2012). Bridging the gap in digital art preservation: interdisciplinary reflections on authenticity, longevity and potential collaborations. In: Konstantelos, L., Delve, J., Billeness, C., Baker, D. and Dobreva, M. (eds.) Preservation of Complex Objects: Volume 2: Software Art. Series: The preservation of complex objects . JISC, pp. 71-83. ISBN 9781861376312. *http://eprints.gla.ac.uk/75165/*.

John, J. L. (2012). Digital forensics and preservation. *Digital Preservation Coalition*.

Joppa, L. N., McInerny, G., Harper, R., Salido, L., Takeda, K., O'Hara, K. & Emmott, S. (2013). Troubling trends in scientific software use. *Science*, *340*(6134), 814-815.

Kenney, A. R., McGovern, N. Y., Botticelli, P., Entlich, R., Lagoze, C., & Payette, S. (2002). Preservation risk management for web resources. *INFORMATION MANAGEMENT JOURNAL-PRAIRIE VILLAGE-*, *36*(5), 52-61.

Kifer, M. (2008). Rule interchange format: The framework. In *Web reasoning and rule systems* (pp. 1-11). Springer Berlin Heidelberg.

Kirschenbaum, M., Ovenden, R., Redwine, G., & Donahue, R. (2010). Digital forensics and born-digital content in cultural heritage collections.

Klutke, G.; Kiessler, P.C.; Wortman, M.A. "A critical look at the bathtub curve". *IEEE Transactions on Reliability* **52** (1): 125–129. doi:10.1109/TR.2002.804492. ISSN 0018-9529.

Kulovits, H., Kraxner, M., Plangg, M., Becker, C., & Bechhofer, S. (2013). Open preservation data:

Controlled vocabularies and ontologies for preservation ecosystems. *Proc. IPRES*, 63-72.

Lagos N., Waddington S., Vion-Dury J-Y. (2015). On the Preservation of Evolving Digital Content – The Continuum Approach and Relevant Metadata Models, Proceedings MTSR 2015, to appear.

Laurenson P. (2005). The Management of Display Equipment in Time-based Media Installations, Tate Online Research Journal. Web. Accessed 3rd Sept. 2015.
*http://www.tate.org.uk/download/file/fid/7344*.

Lawrence, G. W., Kehoe, W. R., Rieger, O. Y., Walters, W. H., & Kenney, A. R. (2000). *Risk Management of Digital Information: A File Format Investigation*. Council on Library and Information Resources, 1755 Massachusetts Avenue, NW, Suite 500, Washington, DC 20036.

Lee, C., Kirschenbaum, M. G., Chassanoff, A., Olsen, P., & Woods, K. (2012). BitCurator: Tools and Techniques for Digital Forensics in Collecting Institutions. *D-Lib Magazine*, *18*(5), 3.

LTDP Earth Observation Guidelines (2012), LTDP Working Group. Web, 30th June, 2012. Web. Accessed 3rd Sept., 2015.
*http://earth.esa.int/gscb/ltdp/EuropeanLTDPCommonGuidelines_Issue2.0.pdf*.

McHugh, A., Innocenti, P., Ross, S. (2008). "Assessing risks to digital cultural heritage with DRAMBORA". International Documentation Committee of the International Council of Museums (CIDOC) 2008, Athens, Greece, 15–18 September 2008.

Media Art Notation System (2013). Web. Accessed 3rd Sept., 2015.
http://www.bampfa.berkeley.edu/about/formalnotation.pdf .

Metrics Based Appraisal project, (2014). University of Illinois.
*https://www.uillinois.edu/cio/services/rims/about_rims/projects/metrics_based_reappraisal/* .
Web. Accessed 3rd Sept., 2015.

Mitzias P., Riga M., Waddington S., Kontopoulos E., Meditskos G., Laurenson P., Kompatsiaris I. (2015). An Ontology Design Pattern for Digital Videos, Proceedings ISWC 2015, to appear.

Smith, M., & Moore, R. W. (2007). Digital archive policies and trusted digital repositories.

Motwani, M. C., Gadiya, M. C., Motwani, R. C., & Harris, F. C. (2004). Survey of image denoising techniques. In Proceedings of GSPX.

Norden, P. V. (1958) "Curve Fitting for a Model of Applied Research and Development Scheduling," *IBM J. Research and Development,* vol. 3, no. 2, pp. 232-248, July 1958.

Norden P. (1970) Using tools for project management. Management of production. Penguin.

Norden, P. V. (1977), *Project Life Cycle Modeling: Background and Application of the Life Cycle Curves.* U.S. Army Computer Systems Command, 1977.

Panagiotis Mitzias P., Riga M., Waddington S., Kontopoulos E., Meditskos G., Laurenson P. and Kompatsiaris I. (2015). An Ontology Design Pattern for Digital Video, Proceedings SW4CH 2015, to appear.

Paradigm project (2008). Web. Accessed 3rd Sept., 2008.
*http://www.paradigm.ac.uk/workbook/appraisal/appraisal-issues.html*.

PERICLES Deliverable D5.1 (2014). Web. Accessed 3rd Sept., 2015. *http://pericles-*

*project.eu/deliverables/12*.

Pillai, K. and Nair, V.S.S. (1997) "Statistical Analysis of Nonstationary Software Metrics," *J. Information and Software Technology,* vol. 39, no. 5, pp. 363-373, 1997.

Putnam, L. H. (1978) "A General Empirical Solution to the Macro Software Sizing and Estimation Problem," *IEEE Trans. Software Eng.,* pp. 345-361, July 1978.

Raz, T., & Michael, E. (2001). Use and benefits of tools for project risk management. *International journal of project management*, *19*(1), 9-17.

Rhizome Collection Management Policy, (1999). Web. Accessed 3rd Sept., 1999. http://rhizome.org/artbase/policy.

Risse, T., & Peters, W. (2012, April). ARCOMEM: from collect-all ARchives to COmmunity MEMories. In *Proceedings of the 21st international conference companion on World Wide Web* (pp. 275-278). ACM

Ross, S. (2012). Digital preservation, archival science and methodological foundations for digital libraries. *New Review of Information Networking*, *17*(1), 43-68.

Sandborn, P. (2007). Software obsolescence-Complicating the part and technology obsolescence management problem. *IEEE Transactions on Components and Packaging Technologies*, *30*(4), 886-888.

Sierman B. et al (2014), "Catalogue of preservation policy elements", SCAPE Project deliverable, *http://www.scape-project.eu/wp-content/uploads/2014/02/SCAPE_D13.2_KB_V1.0.pdf.*

Stamatelatos, M. (2000). Probabilistic Risk Assessment: What Is It And Why Is It Worth Performing It?. *NASA Office of Safety and Mission Assurance*, *4*(05), 00.

TRAC (2007) *Trustworthy Repositories Audit & Certification (TRAC): Criteria and Checklist*. (Chicago, IL: Center for Research Libraries; Dublin, OH: OCLC Online Computer Library Center). *http://www.crl.edu/PDF/trac.pdf.*

Vermaaten, S., Lavoie, B., & Caplan, P. (2012). Identifying threats to successful digital preservation: The SPOT model for risk assessment. *D-Lib Magazine*, *18*(9), 4.

Vieira, R., Ferreira, F., Barateiro, J., & Borbinha, J. (2014). Data Management with Risk Management in Engineering and Science Projects. *New Review of Information Networking*, *19*(2), 49-66.

Zheng, L. (2011). Knowledge representation and decision support for managing product obsolescence.

# 1. Appendix: Entity registry model repository demonstrator

The Entity Registry demonstrator is deployed on a GWDG server: https://c102-086.cloud.gwdg.de

It is using several existing tools: Apache is used to provide a web server, UWSGI manages WSGI applications written in Python, the metadata catalogue is implemented as Cassandra tables and AllegroGraph provides the Triple Store and the SPARQL interface.

In order to demonstrate the provided API, a minimal web interface allows browsing the content of ERMR. The web interface uses the underlying ERMR APIs.
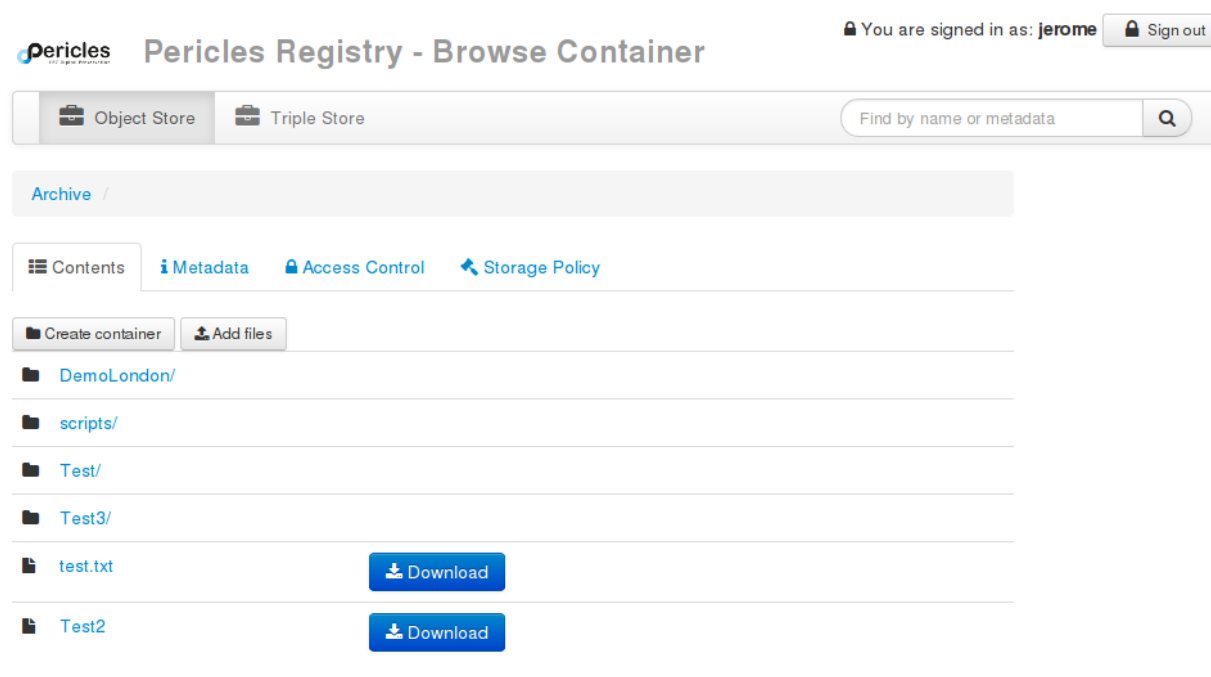


**Figure 22: Main view of the object store**

Figure 22 depicts the main view of the object store of the ERMR. It displays the different sub-collections and the digital objects uploaded to a specific collection. The different tabs (Metadata, Access Control, Storage Policy) can be used to associate metadata or ACL to a collection or a digital object.

The next figure (Figure 23) shows the interface for object metadata. The CDMI URL is a metadata field that is automatically generated. It is a unique identifier for each digital object and it can be reused in the triple store to link an entity to an actual object implementation.

**Figure 23: Metatadata view for objects**

Figure 24 shows the main view for the fact store of the ERMR. It displays a list of triples added to a repository managed by the system. It is possible to execute a SPARQL query to select some triples from the store.



**Figure 24: Main view of the fact store**

# 2. Appendix: ERMR and its role in support of the LRM Services

This section presents a brief overview of ERMR and its role in support of the LRM Services, much of which is set out in Chapter 3 above. This is followed by a more detailed description of the components and the architecture. A diagram of the architecture is included in Section 3.2 above, and is not repeated here.

## Discussion overview

The ERMR is a database in which the data generated by the Linked Resource Model (LRM) Service is stored and transmitted. It consists of a set of components, methods, and usage conventions that provide a way of consistently identifying entities (e.g., digital objects) in the ecosystem, as well as a store of corresponding metadata, model resources, and dependencies.

Acting as "middleware", the ERMR provides a uniform interface to the Linked Resource Model and negotiates connections between the LRM Service and the repositories that constitute the test beds. The ERMR supports synchronisation between the preservation models and the ecosystem; and it provides a *listener*, which can be used to trigger responses to changes in the models and/or ecosystem entities.

The ERMR will negotiate services between the test beds and the project's LRM service, which will be used to execute the ReAL change management language in support of the D3.4 deliverable, and will also be used to activate specific LRM services: for example, to track or control the evolution of the target ecosystem, or track the evolution of LRM ontologies.

Within the context of the overall project, the ERMR forms the basis for querying and storing process models for the *process model compiler,* which creates different models and executable workflows from abstract models; and the *workflow engine* which is responsible for executing the workflows that the model compiler has created; and the *notification system* which schedules communication. The process compiler and workflow engine descriptions are described in the documents forming the 6.4 deliverable.

In negotiating services, ERMR invokes a transaction manager, used by the LRM service to call external services and to control (in a consistent manner) all transformative actions requested on the environment. This transaction manager will operate on local copies of files downloaded from the ERMR, and upload these back to the ERMR when committed, with an updated Universally Unique Identifier (UUID) in the case of modification requests, for example.

## Component overview

We now discuss the specific components constituting the ERMR that support the operation of the LRM Services and their negotiation with the test-bed data stores. The sections include:

- Storage and Content;
- Identification System;

- Query Mechanism;
- Notification / Trigger Functions; and
- Access

## Storage and Content

The ERMR is the *underlying store* for persistent information used to manage the meta-description of entities and related information. Content can be archived *in situ* rather than being moved to a third party archive.

Meta descriptions can include configuration data, references to data objects, information about which interfaces need to be accessed, permissions, XSD schemas, etc., but not the data itself. The data are typically held in external data stores, which are accessed by the interoperable workflows. The implementation strategy in multiple stages enables us to assess the utility of LRM and ecosystem model as part of loosely coupled peer-to-peer federation environments with *external stores.*

\* The ERMR uses *persistent state information* to record all attributes that are needed about a file, including the name of the file, the location of the file, the owner of the file, a file checksum, and data expiration data, and other attributes.

\* The ERMR consists of a *metadata catalogue* that is based on the Apache Cassandra distributed database management system: this is used as the constraint-based metadata management system needed to manage dynamically-defined relationships between the metadata attributes and the data (whether held in external stores or integrated into the archive).

\* The underlying technology supports the *federation of namespaces*, and the *integration of ontology management with information management*.

## Metadata Management

The ERMR is designed to manage the metadata through a central registry, as follows: The ERMR supports a logical name space onto which metadata attributes can be registered. The logical names are used to create global, persistent identifiers. It is assumed that the project's LRM processes will generate state information that can be mapped onto a logical name space for long-term management. The ERMR, therefore, preserves this logical name space as a collection hierarchy, used as the basis for maintaining and querying the metadata attributes.

## Storage and Description of Entities

The ERMR implements a Cloud Data Management Interface (CDMI) that defines the functional interfaces that applications may use to create, retrieve, update, and delete data elements from the Object Store. The Cloud Data Management Interface is an industry standard. Metadata attributes can be set on collections and their contained data elements through this interface.

## Identification System

A crucial requirement is an identification system that assigns unique user IDs (preferably a version 4 UUID). The ERMR achieves this by registering these in the Cassandra distributed database (cassandra.apache.org), added either from the web interface or from a CDMI cloud interface, which

is an industry standard (SNIA) defining the interface that applications will use to create, retrieve, update, and delete data elements from the cloud. Cassandra manages internally the asynchronous replication of each update.

### Query mechanism

The ERMR must be able to enable discovery of descriptive metadata and data through a query mechanism. The project requires the support of metadata in terms of attribute-value-unit triplets. Any number of such associations needs to be added for each digital object. A request might be in terms of logical names, or conditional query based on descriptive and system metadata attributes. This functionality is currently under development, but for the first iteration the query mechanism searches the metadata using a web-form and python code using a standardised, high-level web framework (Django). As there is no capacity in CDMI for query operations, our strategy for the next iteration is to use standard URI queries, which can also be used to allow selective queries on tables and fact stores. Beyond the ERMR prototype, in a production implementation it would be necessary to add application-specific indexes using Apache SOLR/Lucene (or alternative) text search engine. It should be noted that, in the present implementation, the search mechanism will not scale well, and will need to be re-examined as soon as feasible.

### Notification / Trigger functions

This is required to support communication between components in a loosely coupled, peer-to-peer environment. ERMR communication calls are managed by the notification (or trigger) function, which can "listen" to activity in external data stores or models, and take appropriate action when there is a create-read-update-delete (CRUD) activity detected, including changes in the model. The notification / trigger function will automatically execute operations in any scripting language to meet the management policy requirements. In the interests of standards conformance and noting its widespread use by management suites, the initial implement uses a long-established standardised logging (syslog) mechanism, formalised as IETF RFC 5424. We are in the process of implementing a generic local system based on scripts stored in the ERMR itself, with filter patterns stored as metadata on the scripts or their container, with the scripts being loaded and executed dynamically. Further discussion is set out in "Third Party Access Controls" below.

### Logging/auditing/access

Data stores come in a variety of flavours, with various levels of internal structuring such as containers and support for metadata (a.k.a. named properties or key-value pairs), and authorisation mechanisms. We wanted to use to non-proprietary standards, and to present a full-featured object store, and hence we chose to use the CDMI access protocol (which uses classic bearer authentication via http).

### Third Party Access Controls

We operate under the assumption that most remote data stores will have stringent management requirements that mandate policies for users' roles or access rights. The ERMR will need to track this highly changeable information, which forms an essential attribute of the service.

The ERMR supports this capability through the implementation of an authorisation framework based on user and administrator roles and access rights, in which authentication is supported via the Lightweight Directory Access Protocol (LDAP).

In the ERMR, a LDAP server maintains an internal database that maps authenticated users to roles; all access is managed with respect to the role of a user, and user identity is not used (except for logging of selected events). This function is provided by an ERMR storage abstraction mechanism, which can be configured to provide policies relating to who can access objects, the access rights, and the location at which the objects are stored.

The ERMR implementation uses the Cloud Data Management Interface (CDMI) to implement a directory to contain the names of objects and other containers (including both system and user metadata, access control lists, and policy data. The user is authenticated through the LDAP protocol using a password.

ERMR supports access control capabilities through the concept of user roles, signifying permissions and rights. Roles may have administrative rights and access rights to stored objects. As with iRODS and similar data management systems, the permission refers to roles and not users. All access is managed with respect to the role of a user, and the user identity is not used.

The ERMR access controls are managed by the notification (or trigger) functions. Any time in which an event occurs that causes a change in a remote data store, a notification signal is sent out by a standardised message logger (syslog). This will be used to determine whether the notification matches a pattern and will execute an action (e.g. a script), which may further change the repository, send a message, or execute any other action. The ERMR incorporates a listener, which reads python scripts from a default directory and executes them if a notification matches a pattern stored as metadata against the script or its container. Although python is used as the scripting language in the ERMR prototype, the ERMR listener can be configured to execute commands in any scripting language.

## Summary of ERMR functions

| Requirement | ERMR | Notes |
|---|---|---|
| Access to data | HTTP, WebGui | |
| Data Store | Cassandra | |
| External Ref | Via URIs | Can be extended to any URL represented resource |
| Rules | Scripts in any interpreter | |
| Scalability | Peer to peer (unlimited) | |
| Reliability | NO SPOF – all nodes are equivalent | |
| Federation | Rule engine | ERMR mechanisms enable flexible federation schemes |
| Table access | Cassandra tables | |
| HSM | Via custom URI scheme | |
| Performance | Horizontally scalable | |

| | | |
|---|---|---|
| Extensibility | Extensible via common scripting in rules | |
| Access control | Via ACLs | |
| Capacity | Cassandra: Billions of entries, hundreds of petabytes | |
| Rules | Can be added into the repository and executed anywhere | |

**Table 1: Summary of EMR functions**

# 3. Appendix: Policy derivation example

What follows is a short example to demonstrate how policy derivation could be implemented in a real scenario.

*Scenario: an archival institution accepting submissions from publishers.*

**Top-level policy:**

*ID*: TLP-1

*Description:* The archive will be capable of managing the data that is submitted by the publishers.

**Intermediate-level policies:**

*ID:* ILP-1

*Description:* The submission must be in one of approved file formats, and respect constraints

*Details on constraints:*

File size limit **X** has to be respected during online submission.

File format must be from list **Y**.

File must not be copy protected or password protected.

*Context of application:* Digital publications submitted

*Dependencies:*

File format list **X**

Approved file format list **Y**

File format identification tool service

File size limit (possibly per format)

File submission interface to external publishers

Notification system to publishers (ex. email)

**Low-level policies:**

*ID*: LLP1

*Description*: notify publishers of changes in the approved format list

*Statement:*

1 - publish the current supported format list on a public, well known address

2 - notify all new publishers of the list location

3 - notify all registered publishers of any change to the list

4 - notify publisher of list of suggested tools for format identification

ID: LLP2

*Description*: Perform file validation on submitted files

*Statement*:

For each of the incoming or modified submissions

1 - validate file size

2 - determine file format

3 - if (file format is in FileFormat List) return approved

4 - else send (rejected format) to sender

**Concrete process:**

*ID*: CP-1

*Description*: workflow implementing LLP2,

*Link*: URL for the service, or identifier in the system specification

*Dependencies*: CSR1 for file identification

**Concrete services:**

ID: CSR1

Desc: File identification service based on FIDO

Code repository: https://github.com/openpreserve/fido

Version deployed: 1.0

**Unit tests:**

UT-1: validates that file identification service is returning correct file type identifiers for a chosen set of test files; defined on CP1

UT-2: validates that the list of supported formats is accessible from external organisations;

UT-3: validates that the file formats in the list are indeed identifiable by CSR1;

**Scenarios for the QA of policies:**

1. A change in policy mandates the support for a series of new formats. Digging through the dependencies, it is discovered that the current system will not support identification of the target formats (UT-3). For this reason the service is replaced with a new based on DROID.

# 4.   Appendix: Use scenarios for Task 5.3

Policy QA is an important aspect, and examples of how policies can be ignored or changed in practice, are not difficult to find in the news.



**Figure 25: A recent article showing how policy implementation can deviate for technical reasons**

In the example in Figure 255[35], it is shown how a police department unilaterally decided to change a retention policy as consequence of technical difficulties. In this case, it is clear that a model supporting top-down (from principles to technical implementation) can be useful to make principles drive the technical implementation, and not vice-versa.

In general, we forecast that our approach will be useful for a number of cases:

- Understanding the risks and impact associated with a change of policy.
- Being aware of changes (external and internal) that may invalidate a policy implementation.
- Understanding and controlling how the content of the archive may evolve over time.

When a standard checklist of policies needs to be implemented to adhere to a standard, the policy derivation and mapping will help prove the correct implementation. The policy model can assist in a number of scenarios:

- As an archive manager, I want to understand the risks and impact associated with a change of policy.
- As an archive manager, I want to be made aware of changes (external and internal) that may require a change of policy.
- As a data user/owner, I want to understand what policies are applied to these DOs
- As a data owner or archive manager, I want to understand and control how the content of the archive may evolve over time.
- As an archive manager, I want to monitor the usage to determine whether policy changes are required to meet the (changing) user needs.
- In order for my archive to be approved for use it has to abide by a standard checklist. These checklist items need to be turned into a set of procedures. I need to be able to create these procedures and to demonstrate that they implement the checklist.
- As a data owner I want my personal policies regarding my content to be regarded by the archive.

---

[35]  http://arstechnica.com/tech-policy/2015/08/cops-decide-to-collect-less-license-plate-data-after-80gb-drive-got-full/

# 5.  Appendix: Appraisal factors

Note: This extends and adapts DELOS 4.3 (Guercio et al, 2008)

| Category | Appraisal factor | Example | Metrics | Aggregation level | Usage |
|---|---|---|---|---|---|
| Content | Comprehensiveness | Does information cover a complete population? Does data collection include all relevant parameters (for example: is data from SOLAR sensor provided alongside data streams used for configuration?) | Coverage; statistical validity (cf. significance); statistical representativeness | Item, series, collection | Arts; Science data Geospatial data[1] |
| Content | Content-level change | Is information subject to on-going processes of change (for example: review and refinement, renormalisation?) | | Item, series | Science data Arts |
| Content | Descriptive metadata | Is there sufficient metadata to interpret the data? | Is classification data provided? Is descriptive metadata provided? | Item | Science data |
| Content | Growth | Will information continue to grow or is object complete? Is this object part of an on-going collection? | Time-based review requirements Periodic maintenance requirements Conformance to expected timescales Periodicity/frequency of data capture | Item, series | Arts; Science data; Geospatial data[1] |
| Content, Contextual | Relationships, links, interdependencies | Relation to current and future research Relation to art gallery, installation sequence Relation to other material held within dataset/collection | Relationship to other items; links with broader ecosystem; citation network, etc. | Item, series, collection | Arts; Science data; Social science data[1]; records[1], publications[1] websites[1] geospatial data[1] |
| Content | Significance | Policy significance; operational significance; functional significance | Considerations of dependencies or interdependencies Use of items as finding aids/proxies for other items Provenance | Item, series | Science data,; Arts; Social science data[1]; records[1], geospatial data[1], |

| | | | | | websites[1] |
|---|---|---|---|---|---|
| Content | Reliability | Whether the information is likely to be accurate or authoritative. 'Contents can be trusted as a full and accurate representation of the transactions, activities or facts to which they attest' (ISO15489) | cf. integrity, functionality, stability<br><br>Statistical reliability of data; associated uncertainties and error margins | Item | Science data; Geospatial data[1,] publications[1] |
| Content | Spatial coverage | The spatial area covered | Geographic coordinates<br><br>Bounding box | Item | Science data; Geospatial data[1] |
| Content | Temporal coverage, time, duration | Period of time covered, e.g. creation date and end date. | Time period covered<br><br>Creation date<br><br>End date<br><br>Sampling rate(s)<br><br>Periodicity/frequency of data capture (cf. 'Growth') | Item, Series | Arts; Science data; Social science data[1]; records[1]; geospatial data[1] |
| Content | Uniqueness | Does the resource represent unique information, or is the object available elsewhere? | Do duplicates exist?<br><br>Is information available in other media? | Item, series | Arts; Science data; Social science data[1]; records[1]; publications[1], geospatial data[1], websites[1] |
| Content | Usability | Accessibility of content, e.g. are appropriate manuals available to decipher information | Accessibility of object<br><br>Ease of use<br><br>Accessibility/readability of documentation | Item, series | Arts; Science data; Geospatial data[1]; social science data[1], records[1]; websites[1] |
| Content | Version | Which version of the material is represented? | Which versions exist?<br><br>How are versions controlled? | Item | Science data |
| Contextual | Authority | Authority of object | Reuse within holding organisation<br><br>Reuse by research community<br><br>Use in peer-reviewed material<br><br>Use in non-peer-reviewed material | Item, series | Scientific data; art; written text |

| Contextual | Conformance to standards | Does the object conform to relevant standards or guidelines? | Checklist comparison where relevant<br>Validation where relevant | Item | Scientific data |
|---|---|---|---|---|---|
| Contextual | Documentation | Accompanying technical documentation explains how data collected etc. | Documentation of production process<br>Documentation of specific item<br>Documentation of extant copies and related items (cf. 'uniqueness') | Item, series | Social science data[1]; geospatial data[1] |
| Contextual | Intention of artist/creator | Availability of accompanying documentation providing artist's stated intentions and preferences, where available | Approval of installations<br>Declared meaning of object<br>Evidence of artist's values | Item, series | Arts |
| Contextual | Intention of curator | Intention of creator/curator of installation/instance | Archival value for installation<br>Documented outcomes of consultation with experts, curators, creators, researchers | Item, series | Arts; science data |
| Contextual | Impact of object | Evaluated impact of the object<br>Cf. authority<br>Impact relates to written/spoken/inferred 'buzz' (as with academic citations). Authority may relate to a few very important links or identifications, rather than a large number of mentions. | Reuse within community, within peer-reviewed material, within non-peer-reviewed material, references by the media industry | Item, series | Arts; science data |
| Contextual | Installation context | Evaluation of the installation context, including audience experience, curation, maintenance and safety requirements; documentation of physical location and setup<br><br>Sample questions: What is the audience's experience of the item?<br>How difficult is the item to set up and maintain? | Maintenance requirements for the installation<br>Visibility of equipment<br>Auditory, visual, olfactory and kinematic context<br>Physical location<br>Security and safety context<br>Creator/curator of installation | Item, series | Arts; scientific data; performance |
| Contextual | Meaning of object | Stated goal of creator; for example, a dataset may be designed with the intent of | cf. 'Intention of artist/creator' | Item, series | Arts; science data |

| | | | | | |
|---|---|---|---|---|---|
| | | studying a certain phenomenon. This may also touch on the process of reframing. Acquired meanings may relate to post-hoc use/referencing; meanings may be assigned as a result of lived experience | | | |
| Contextual | Provenance | Provenance, history and origin of object | Appropriateness of provenance to collection (e.g. 'within state', or existence of relationship with donor) | Item, series | Arts; Science data; Records; geospatial data[1]; websites[1], publications[1] |
| Contextual | Significance | Significance of source/context of data/records | | Item, series | Arts; Science data; Social science data[1]; records[1] |
| Contextual | Usage | Frequency of use | cf. 'Potential', 'Impact' | Item | Arts; Science data; Geospatial data[1] |
| Evidence | Accountability | Provide defence of agency against charges of fraud/misrepresentation | | Item | Geospatial data[1] |
| Evidence | Associated/derived materials | How technology incorporates into business: Provide evidence of business processes; de facto policy; procedural management; policy compliance – proxy data for social/scientific context | Evidence of installations – modifications Operational context | Item, series | Science data, Arts, Websites[1] |
| Evidence | Authenticity | Whether the object is what it purports to be, to have been created or sent by the person purported to have created or sent it; to have been created or sent at the time purported | Quality control/quality assurance Reliability – is the object full/accurate? | Item | Arts; Science data; Records[1,] geospatial data[1] |
| Evidence | Precedence | Documentation of decisions that set precedent | Cf. Associated/derived materials | Item | Records[1] |
| Operational | Costs | Costs involved in long-term maintenance | Cf. Replaceability; integrity; installation context; growth | Item, series | Arts; Science data; Social science data[1]; |

| | | | | | records[1] |
|---|---|---|---|---|---|
| Operational | Collection policy | Fit with existing collection policy | cf. 'Mission' <br><br> Relevance to targeted community <br><br> Potential for targeted research community <br><br> Extent to which it addresses current or known application needs | Item; series | Arts; science data; Geospatial data[1] |
| Operational | Financial value | Establishment and review of financial value | Insurance costs; replacement cost; existing revenue generated; projected revenue generation | Item; series | Art; science data |
| Operational | Mission | Fit with organizational mission | Relevance of subject/theme to organisational mission | Item; series; collection | Arts; Science data; Geospatial data[1] |
| Operational | Potential | Can the material be repurposed? <br><br> May include legacy versioning and past impact, as well as change tracking operations. | Evaluation of impact; existing patterns of reuse; community interest in future reuse; interest of holding organisation in reuse | Item | Arts; Science data; Geospatial data[1] |
| Operational | Replaceability | Can information be replicated? | cost of replicating information; value of information vs costs of preservation | Item, series, collection | Arts; Science data; Geospatial data[1] |
| Societal | Ethics | Ethical implications that may influence decision making | Reasons why data should not be retained <br><br> Changing landscape alters significance of data, cf. semantic change | Item, series, collection | Any |
| Societal | Historicity | Representativeness within oeuvre; relation to broader cultural or historical context; This is a rescoping of 'representativeness' within a given historical context (e.g. the life of an artist or the subject under discussion) | Evaluation of historical value | Item, series, collection | Arts; Science data |
| Societal | Intrinsic value | Perceived aesthetic or artistic quality, experimental use of new technology. Outstanding aesthetic value or | | Item | Arts; Science data; Research data; Records[1]; geospatial |

| | | | | | |
|---|---|---|---|---|---|
| | | accomplishment. | | | data[1] publications[1] |
| Societal | Perceived value | Extrinsic perceptions of value Scientific value Spiritual value Social value | Personal judgement of relevance | Item, series, collection | Arts, Science data |
| Societal | Legal considerations | Privacy, data protection legislation prohibiting retention; adherence to privacy/confidentiality regulations and best practises | | Item, series | Arts; Science data; Geospatial data[1] |
| Societal | Social representativeness | Representativeness of sections of society; statistical/demographic representativeness | Cf. comprehensiveness, historicity | Item, series, collection | Art; Records[1]; geospatial data[1] |
| Technical | Functionality | The functionality of the object; the look and feel of the object | Has the look and feel been retained? Is the current functionality of the object equivalent to the original functionality? | Item | Art; Websites[1] |
| Technical | Integrity (a.k.a. condition) | The condition of the object should accord with expectations. Expectations vary with policy, field, and judgement of 'significant properties' - expectations. In some electronic records the expectation may be that records remain complete and unaltered. For physical objects it may in some cases be expected that the condition of objects be congruent with the object's history. Note: Functionality is close to integrity 'on an API level' – that is, when an item is treated as a 'black box'. | Accordance with archive policy Evaluation of the effect of decay on the object Integrity – is the object complete/unaltered? Evaluation of physical condition (e.g. mechanical, chemical aging) Damage incurred to object Periodic reviews on condition of object Technical obsolescence | Item | Art; Software; Science; Records[4,], geospatial data[1] |
| Technical | Logical size, content and composition | Logical size, content and composition of object | File format Filesize Endianness | Item | Art, Science data |

| | | | | | |
|---|---|---|---|---|---|
| | | Cf. stability<br><br>This comprises a technical aspect of the overall cost/risk maintenance analysis | Format version & profile | | |
| Technical | Rights issues | Clarity of licencing and IP situation; affordability of ongoing licencing; acceptance of licencing requirements cf. social/legal considerations, cost (preservation model planning) | Accordance with relevant contractual agreements | Item | Art, Science data; Geospatial data[1] |
| Technical | Risk | Degree of risk to content<br>Degree of risk to infrastructure<br>Impact of potential changes<br>Knock-on effects of change (critical component analysis; bottleneck analysis) | Identification and management of relevant security obligations<br>Models of risk; appropriate analyses | Item, series, collection | Art; Science data; Geospatial data[1]; records<br>Social science data[1] |
| Technical | Physical size, content and composition | Physical size of object; physical content; physical composition<br>Physical size of data on drive | Material<br>Size – height, depth, width<br>Weight/mass | Item, series | Art; Science; Social science data[1]; records[1] |
| Operational, Technical, Social | Stability | Can the object, its ecosystem and its interdependencies be viewed as reliable? | Do the policies and funding models of providers, suppliers or related archives permit the object to be viewed as stable? This judgement is likely to be derived during an appraisal process | Item, series, ecosystem | Art; science data; websites; networks |
| Technical | Record availability, discoverability and accessibility | Availability, discoverability and accessibility of records - should be able to be located, retrieved, presented and interpreted<br>Usability in broad sense may impact on this.<br>Cf. integrity, technical obsolescence, semantic change | Discoverability<br>Measures of accessibility | Item | Art; Science; Geospatial data[1]; social science data[1] records[1]; websites[1] |

Notes:

1: The use of this criterion in this application area is recommended for this purpose by DELOS 4.3.