Project Number	IST-2006-033789
Project Title	PLANETS
Title of Deliverable	PLANETS Core Registry: Future Vision Document
Deliverable Number	PC3-D24
Contributing Sub-project and Work-package	PC/3, PA/3
Deliverable Dissemination Level	External
Deliverable Nature	Report
Contractual Delivery Date	31st March 2010
Actual Delivery Date	31st May 2010
Author(s)	TNA, KBNL

Contributors

Person	Role	Partner
Lynne Montague	Author	TNA
Sara van Bussel	Author	KB-NL

Document Approval

Person	Role	Partner
Tim Gollins	Subproject lead	TNA
Barbara Sierman	Reviewer	KB-NL

Distribution

Person	Role	Partner

Revision History

Issue	Author	Date	Description

Other References

Ref.	Document	Date	Version
PC3 D20 V1.R2.M0	PCR - Software Requirement Document	21/12/09	Final
PC3 D20 V1.R1.M3	PCR - Software Requirement Document	16/11/09	Draft

EXECUTIVE SUMMARY

The purpose of this document is to set out a future vision of possible developments for the PLANETS Core Registry (PCR). This is based on both those elements of the PCR which have previously been specified during the course of the PLANETS project as needing further development, for example PUID assignment and the implementation of a faceted classification system, and further ideas for potential development, such as the use of linked data.

The document aims to set out the major areas for potential development at a high level and does not cover the minutiae of every proposed requirement as set out previously during the project. However, elements from the Software Requirements document that were not integrated into the current version of the PCR have been listed in Appendix A in detail, for reference.

TABLE OF CONTENTS

1. Introduction

1.1 The PLANETS Core Registry

As part of the four-year, European- Union-Funded PLANETS project, two new (and one enhanced) versions of what is now known as the PLANETS Core Registry (PCR) have been designed, built, tested, released and populated by the Preservation Characterisation and Preservation Action subprojects[1]. The PCR combines the Preservation Characterisation Registry and the Preservation Action Registry which were developed at earlier stages of the project.

Whilst design requirements for PCR version 3 were produced, it was not possible to undertake the development of the software within the scope of the PLANETS project Therefore the current and final release within PLANETS is PCR version 2.1, a single instance of which is hosted by PLANETS partner, the Humanities Advanced Technology and Information Institute (Hatii), at the University of Glasgow (http://corereg.arts.gla.ac.uk/PLANETSCoreRegistry/welcome.html;jsessionid=C45C3925B2D1DF560D8AC00E978EE96E).

The PCR is a technical registry that stores core records for File formats, Software, Hardware, Compression Techniques, Character Encodings and Storage Media along with associated subsidiary records and reference information. This persistent, unambiguous technical information supports characterisation, preservation planning, preservation action and preservation watch functions and provides a growing source of technical reference information to the digital preservation community[2].

The PCR already plays an important role in the PLANETS suite of digital preservation tools. This role could be improved by future development and fuller integration with other PLANETS services (see sections below). The Open Planets Foundation has been established to provide practical solutions and expertise in digital preservation beyond the life of the PLANETS project, and will use and build on the work undertaken by the PLANETS consortium in providing the PLANETS suite, including PCR2.1 (http://www.openplanetsfoundation.org/).

1.2 Purpose of this Document

The purpose of this document is to set out a future vision of possible developments for the PCR in order to improve both the software and the methods for storing information. This is based on both those elements of the PCR which have previously been specified (as part of the design for PCR 3) as needing further development (section 2), and further ideas for potential development and improvement (section 3). The document aims to set out the major areas for potential development at a high level and does not cover the necessary organisational aspects which would be involved in implementing them, nor the minutiae of every proposed requirement as set out in the draft Software Requirements document for PCR3. However, the elements from that document that were planned for PCR3, or further future development, and were not implemented in PCR 2.1, are listed in Appendix A in detail.

2. Future developments – the design for PCR 3

The suggestions for future developments within the following section are based on the high-level ideas identified in the design for PCR 3 and set out as detailed requirements in Appendix A. It will cover facetted classification, the assignment of unique identifiers and their uniqueness, technology watch alerts, usability, integration with the PLANETS suite, audit trails, rollback and version control.

[1] These have been developed as enhancements of the PRONOM Technical Registry developed by The National Archives of the UK (TNA). Originally known as the Preservation Characterisation Registry, it became known as the PCR when it was combined with the Preservation Action Registry.

[2] For a more detailed description of the role the PCR plays within the Planets project see Sinclair, P. (2009). *Core Registry V3: Software Requirements Document.* Retrieved from http://www.planets-project.eu/docs/reports/Planets_PC3-D20_Software_Requirements.pdf. Planets deliverable PC3 – D20.

2.1 GDFR facets

One of the requirements for the PCR[3] was the introduction of an extensible faceted classification scheme for file formats in accordance with that proposed by the Global Digital Format Registry (GDFR)[4]. The intention was that this more extensible system should replace the existing Entity Type and Entity Family classifications within the PCR, which were inherited from PRONOM, and provide alignment with the Unified Digital Formats Registry (UDFR). The idea behind the UDFR is a move towards a single shared formats registry[5].

It was noted that the GDFR scheme might require minor modification to meet the needs of PLANETS and that further work would be necessary to pinpoint the precise scheme. Whilst the GDFR scheme is implemented in PCR 2.1, further work still needs to be undertaken to clarify and develop its usage.

The GDFR uses a system of facet type/value pairs, with eight main and one subsidiary facet specified. It is anticipated that additional facet type/value pairs will be necessary to meet the classification requirements of the PCR and further work will be needed to identify these and expand the faceted classifications available. For example, there is currently no genre classification for animation or virtual reality.

It is currently possible to associate single or multiple facet type/value pairings to an entity incorrectly. The only control on this would be the expertise of the administrator to apply the classifications correctly. It would be desirable in the future to build rules into the PCR to ensure correctly applied associations between entities and facets.

Although it is specified within *Guidance for populating characterisation elements*[6], that Entity Families will no longer be used within the PLANETS project, they have not been totally removed from the system in order not to lose existing information. However, very little information has been recorded for Entity Families in previous version of the PCR and in fact the only types of Entity Family that can be recorded are for file formats. Therefore, the consequences of removing the Entity Family classification for future versions of the PCR should be investigated and, if no problems are identified, the classification should be removed from the PCR.

2.2 PUID Assignment

A scheme of persistent unique identifiers (PUIDs) has been implemented within the PCR[7]. This extensible scheme provides persistent, unique and unambiguous identifiers for information recorded in the PCR. A PUID has two parts: the PUID type, which identifies the class of information being referred to, e.g. file formats, and a numeric identifier which is unique within that class.

PUIDs are fundamental to the exchange and management of digital objects because they both supply an inherently unique identifier, and bind this to a definitive description from the PCR of the object being identified. These identifications can be used by both human agents (as reference objects) and automated agents (when passing digital object information between the components of a digital preservation system), to unambiguously identify, and share that identification of, classes of information.

Vital to the use of PUIDs are:

- Uniqueness: Each PUID must be unique to a single unit of information, such as a specific version of a file format.

[3] As specified in internal Planets deliverable PC3- D4 *Registry Iteration 2 Design - User Requirements Report*

[4] http://www.gdfr.info/docs/GDFR-Classification-1_0_5.pdf

[5] A format registry working group has been set up by international organisations in order to build a community, and solicit requirements with the aim of establishing such a registry.

[6] Planets deliverable PC3-D12: *Guidance for populating characterisation elements.*

[7] This scheme is based on one implemented by TNA for it's Pronom registry and is described fully in *The PRONOM PUID Scheme: A scheme of persistent unique identifiers for representation information.* Retrieved on 3rd May from http://www.nationalarchives.gov.uk/aboutapps/pronom/pdf/pronom_unique_identifier_scheme.pdf

- Persistence: Once assigned, PUIDs must be persistent. As such, they must be immune from changes in technology or scheme administration and they must be sufficiently flexible to be adaptable to future developments without any need for changes to existing identifiers.

2.2.1 Ensuring PUID Uniqueness

Within the PCR, file formats, software packages, hardware, character encoding, compression techniques, storage media, technical environments, pathways, and properties are all assigned PUIDs. Currently, the uniqueness of a PUID is ensured by each instance of the PCR software. As each entity requiring a PUID is entered into the PCR, a PUID is chosen by the administrator. If this duplicates a previous PUID the PCR notifies the user that the PUID is already in use and will not allow the entry to be saved. As there is only one, central instance of the PCR within PLANETS, the uniqueness of PUIDs can be guaranteed during the course of the PLANETS project. However, it is possible that in the future, multiple synchronised instances of the PCR would be desirable and in that situation, systems would need to be implemented in order to ensure that PUIDs are not duplicated.

One option is for number ranges to be allocated for each type of PUID for each instance of the PCR. The scope of the PCR software does not currently allow this to be done within the software itself so human policies would need to be implemented to specify this allocation. In this way, unique number ranges for each entity can be allocated to each instance or alternatively, a particular instance may not be allocated any numbers for a specific entity and therefore will have no new instances of that entity entered into it. This would then limit the responsibility for entering information about certain entities to administrators for specific instances only. However, this is merely a suggestion based on discussion during the course of the project and it is recognised that there may be more sophisticated solutions to this problem.

2.2.2 Allocation of PUIDS

Based on previous discussions within the project, it is envisaged that there would be two ways of allocating the numeric part of PUIDs in future versions of the PCR:

- A function within the software that allocates the next available entity type–specific PUID to an entity from within the allocated ranges. If there are no unused PUIDs left within a range or the range is set to zero, it would not be possible to create a new PUID.
- An option for an administrator to be able to enter a PUID from within the allocated ranges. This would be associated with a function which checks the entered PUID for uniqueness within that particular instance of the PCR and would not allow the entity to be saved if the PUID is a duplicate.

If the user enters a PUID that is not within an allocated range, the PCR would warn them of the fact but allow them to override this and continue to save the new entity. In this way, PUIDs that have been assigned to an entity in one instance of the PCR can be manually duplicated in another to ensure that the entity has the same PUID in each. However, if synchronisation between instances of the PCR were enabled, this would be unnecessary.

2.3 Technology Watch Alerts

A technology watch function has been set up within the PCR in order to monitor the registry, to identify changes to data that may influence preservation planning decisions and advice and to send out alerts to notify subscribers of these changes.

Currently these alerts can be generated, via email, for creation, modification and deletion changes to limited, specified fields within the six core entities of File Format, Software, Hardware, Storage Medium, Character Encoding and Compression Technique. A simple suggestion for the future is to extend the types of data monitored under this system and to generate the alerts in a machine-readable form.

For example, alerts could be generated for changes to:

- Software package tools and processes;

- Withdrawn details i.e. when support is no longer available for a particular entity;
- File format risk scores and properties;
- Technical environments;
- Testbed information associated with a preservation pathway step.

A more major development would be to fully integrate this service, in an automated way, with a preservation planning process, such as the PLANETS Preservation Planning Tool, Plato. In this way, identified changes to data could be submitted as a parameter of the preservation planning process and a pro-active preservation planning advice service could analyse these technology watch alerts, and changes to institutional policies, in the context of a specific collection profile (also a parameter of the preservation planning process). A preservation plan evaluation process could then be executed.

However, the technology watch function is a visionary concept that assumes, and is dependent on, sufficient resources to maintain and update the PCR on a continual and ongoing basis. Without this level of population of the PCR, very little information will be generated to inform the preservation planning process. It is unclear as yet whether this assumption of context holds true and thus, whether the technology watch function is viable.

2.4 Usability

It has not been possible within the scope of the project to fully assess and evaluate the PCR in terms of human usability i.e. the design and functionality of the system and the ease with which the human interface can be used. One of the requirements for the system is that it should be intuitive and easy to use without training and it is felt that further work could be undertaken to improve this aspect of it. For example, user documentation to provide guidance for the system has been written as part of the Planet's project[8] but a future iteration of the PCR could provide a greater degree of help with defining terms and guiding usage, from within the system itself.

2.4.1 Search Interface Requirements

The PCR search interface allows users to search the repository, view the results of these searches and see detailed reports about the core entities held within the repository. Currently, users are able to search by text, by PUID or by pathway. However, it was not possible within the scope of the PLANETS project to develop the search functionality to the level it would have been liked. Therefore refinements to the searching process are suggested for greater ease of use, in order to produce more valuable results and in order to display the results in a more informative way. For example:

- Advanced Software searching. Discussions during the PLANETS project highlighted the need for the user interface to allow users to carry out an advanced search for software packages, based on key fields such as those specified in Figure 1 below. These extra fields enable more precise searching based on available information to differentiate between similar versions of software packages. This would also enable searching for Software Packages based on the file formats that they can work with (create, render, edit etc.).

8 Internal Planets deliverable PC3-D16: *User documentation for PCRv2.1*

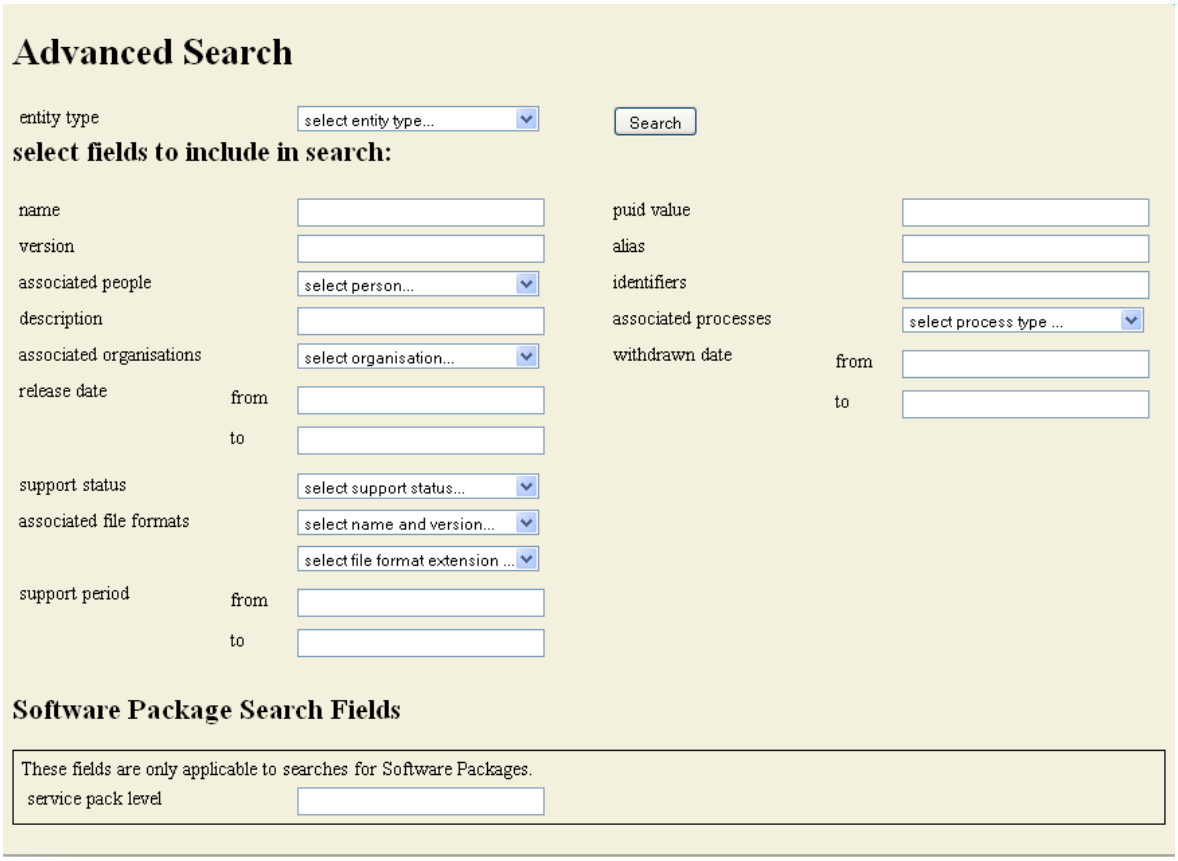

Figure 1: Advanced software search fields

- Advanced File Format searching. As with the search interface for the TNA's Pronom registry, it would be advantageous to be able to do a more detailed file format search that allows the user to search not only for formats, but also for software packages capable of processing file formats. Processing means creating, rendering, identifying, validating or extracting metadata from a file format.

  In addition, searching for file formats (either generally or on the basis of file format families), based on their inherent risk scores could also be enabled. For example it would enable the user to search for image or presentation formats with low preservation risk scores in order to inform preservation plans within an organisation with a low risk appetite.

- Pathway searching. Currently, this search function is dependent on knowing and entering the PUIDs for the source and target file formats concerned. Enabling this search to also be done based on file extensions or file format names would lead to increased ease of use.

- PUID search. The current system is dependent on entering the full PUID i.e. the entity-specific three-letter identifier with a slash plus the numeric identifier e.g. fmt/123. It would be easier and quicker if it was possible to omit the entity type code and the slash so that only the number needed to be entered e.g. 123.

2.5 Full integration with the PLANETS Testbed

Within Planets a Testbed was developed to perform objective tests on preservation action tools. The results from these tests are accessible from within the Testbed itself. However, the plan was to integrate these test results into the PCR, in an aggregated form, to make them easily accessible to other Planets tools (such as Plato) and outside parties, both through browsing and web services. The test results would be placed at a pathway level in the PCR, giving information about specific

services and interactions.

With these aggregated results the PCR could provide insight into how tools perform in various pathways. At a higher level, another possible interaction would be for Plato to tap into the Testbed information in the PCR to match that information to weighed criteria in order to present a user with the best possible tools and actions for his preservation needs.

Because of the simultaneous development of the PCR and the Planets Testbed, full integration of the two has not been possible within the scope of the PLANETS project. During a preliminary meeting in which ideas were exchanged about this integration, it became clear that some work needs to be done on the models underlying both the PCR and the Planets Testbed to enable full integration to be workable.

The interpretation of Pathways as used by the PCR (one or more tools to perform a preservation action with one input file format and in the case of migration one output file format) is slightly different from the pathways tested in the Testbed. For one, the tests in the Testbed include characterisation (before and after), which the PCR cannot accommodate. There are more such differences.

If an agreement is reached on this difference of models, and the Testbed test results can be linked or imported one-on-one to the PCR, the manner of this link or import needs to be decided. An automatic import of test reports into a quarantine area has been discussed. There reports would then need to be viewed by an administrator (to make sure they are for serious testing) and if approved, would be linked to the correct area in the PCR. These linked test results would have to be returned via web services if the Pathway or tool is requested by an external user.

2.6 Full integration with Plato

The preservation planning service Plato helps users write a preservation plan for their digital collection. During the workflow Plato presents information to the users about file formats and preservation action software. This information is currently gathered from various existing databases.

In the future this information could be gathered using the PCR. The Plato tool could retrieve this information in the PCR using web services and this would enable Plato to present the information to its users in a uniform manner without having to guide them to the PCR.

2.7 Audit trail

The PCR strives to maintain a high standard of information. The process of gathering and entering information should be as transparent as possible to heighten the trust users will have in the PCR. One of the ways to accomplish this is to make an audit trail available to users of the PCR. In this audit trail users can see what information has been added, changed or deleted and by whom. Users should also be able to compare versions of a record. Another feature would be for users to be able to view a record as it was on a certain date. This can help them understand why certain decisions have been made in the past while current information might point to a different decision.

There is currently a rudimentary audit trail available to administrators of the PCR. In this audit trail information can be viewed about who created, edited or deleted a record from the PCR, with a timestamp. These audit records are available from each entity record (for that record), and as a chronological listing.

However more information for this audit trail could be provided. For instance, there is no information available about what exactly the edit entailed (which field and which data, before and after) and this information could be available in the audit trail. There could also be a possibility to add comments about the specific change that was made to a record. This information is especially crucial when there are multiple administrators.

2.8 Rollback and version control

Together with a more advanced audit trail, the ability for rollback and version control should be included. This is directly related to the functionality described above. This functionality, currently not present in the PCR in any form, will enable administrators to roll a record back to an earlier version after an erroneous edit or delete action. This means that all versions of a record will remain

available to administrators. This will also enable users to view earlier versions of a record and compare those with current versions.

3. Possible future scenarios

This section details possible future scenarios by drawing on two major areas influencing current modes of thinking and development within the field of digital preservation; technical registries and linked data. It is not intended to be a definitive statement about all possible directions for development.

3.1 Multiple registry instances

3.1.1 Registry Networks

As stated above in section 2.2 it is possible that in a future scenario, there may by multiple instances of a format registry such as the PCR, sharing data. The structure of these registries and their interactions with each other could take a variety of forms. One possibility is for there to be a peer-to peer network of registries, which share the same information. Having a distributed architecture such as this leads to increased scalability, where new registries can be added (or taken away) on an ad-hoc basis, and service robustness, where the removal of one of the registries does not significantly impact the network i.e. there is no single point of failure[9].

As stated above, there is currently only one instance of the PCR which is hosted at Hatii, with PLANETS partners The National Library of the Netherlands/Koninklijke Bibliotheek (KBNL) and The National Archives of the UK (TNA) responsible for inputting data. However, an alternative would be to have two or more instances based at other of the PLANETS partners.

Within such a network there are many possibilities for how the individual instances could be organised. One possibility is for administrators for each instance to have distinct responsibilities for entering data. For example one organisation might have an instance which it populates with information relevant to preservation characterisation services e.g. file format, property and compression information, whilst another organisation might have an instance which is populated with information relevant to preservation actions e.g. software, hardware and pathway information. Any overlap of information to be entered, for example documentation, would need to be managed externally and manually, being covered by agreements between the different organisations. Periodically the separate instances would be synchronised, so that all of the information is made fully available to users of both instances.

Within this structure, protocol could dictate that information regarding preservation characterisation would be mastered by organisation one and likewise for preservation action information for organisation two. If organisation one leaves the network and is replaced by organisation three, the protocol could be changed so that they are then given the responsibility for producing master preservation characterisation information.

3.1.2 Registry Hierarchies

Another option is for a hierarchy of registries to be established, with information passed down through the hierarchy from top to bottom. For example, through collaboration with international institutions, an international registry could be developed[10]. In this way there could be a global instance holding common file format and process data with local, domain-specific instances taking data from the global instance and adding their own data, which is then passed on to organisation-specific instances which may also contain their own system-specific information. These separate instances could exhibit different levels of sharing e.g. information could be organised into what is public and what is policy-, organisation-, or sector-specific (and therefore 'private' to at least some level) for that instance.

[9] http://en.wikipedia.org/wiki/Peer-to-peer

[10] For example the work being undertaken for the UDFR, as mentioned in section 2.1 above.

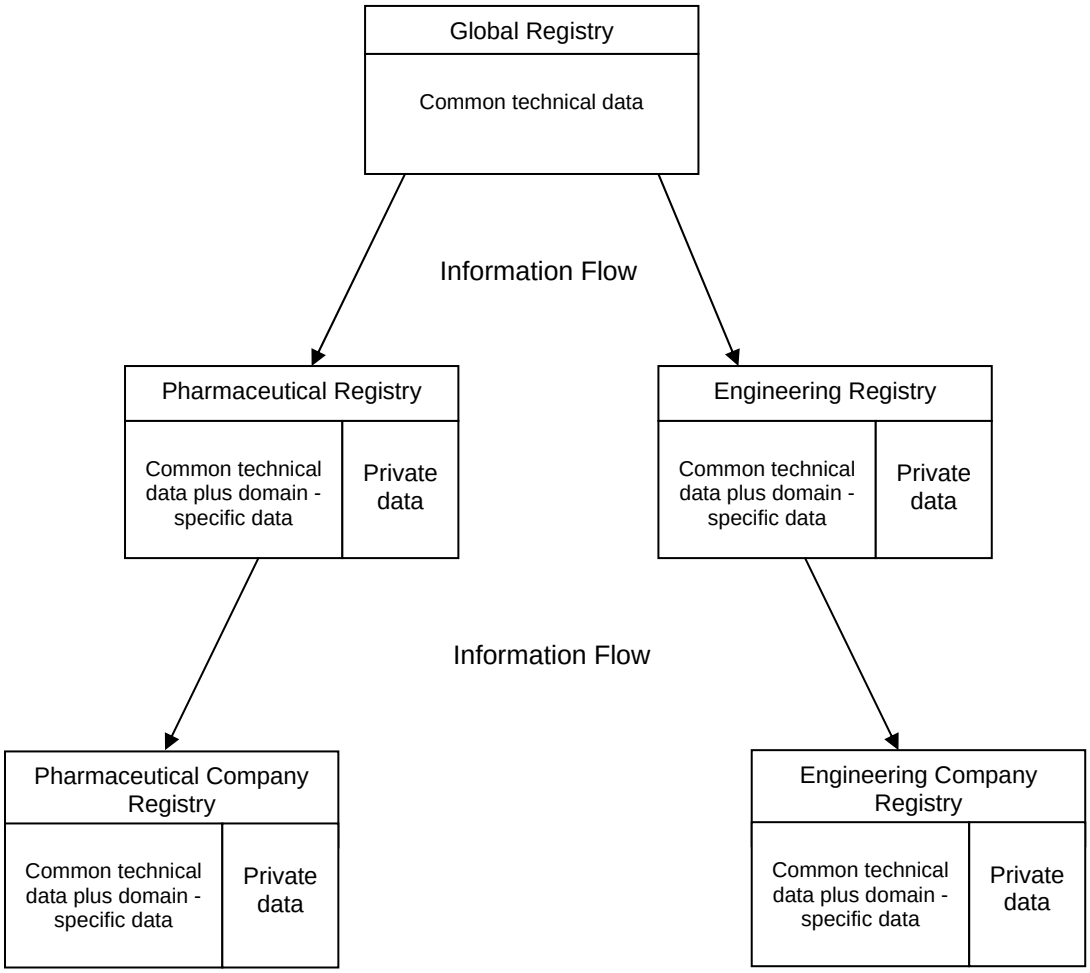

Figure 2: example of a potential registry hierarchy

This hierarchy would rely on the registries being able to pass on both total and partial copies of the data held within them, as appropriate, and also being capable of only overwriting appropriate data when updates are received from a registry higher up in the hierarchy. This is particularly important because registries such as the PCR can be used in two ways. Firstly, they can be used as a passive source of technical reference material that is useful to the greater preservation community and secondly, they can be used in an automated way to inform and control digital preservation systems by, for example:

- Providing signature files to enable format identification;
- Listing identification tools;
- Listing property extraction tools and the properties they can extract, for a given format;
- Providing xml schemas to enable xml validation;
- Listing validation tools for a given format;
- Providing information on the inherent property risk for a given format's property value
- Listing formats with risk higher than given threshold
- Listing properties that should be measured

- Listing migration pathways given source and optional target formats
- Setting a pathway to be the default pathway for its source format

In the latter case, when a registry is used as a controlling instance for a preservation system, this capability of the master, or higher level, registry to only overwrite certain information becomes vitally important. For instance, in the example in figure 2 above, the Pharmaceutical Registry may contain domain-specific migration pathways which have been agreed in conjunction with a regulatory or government body such as the U.S. Food and Drug Administration. In this situation it is vitally important that any generic migration pathways, received in updates from the global registry, do not overwrite these pharmaceutical-, domain–specific migration pathways. At the same time, the Pharmaceutical Registry may contain 'private' information that is not appropriate to be passed on to individual company registries and therefore only a partial transfer of information would be desirable here too.

3.2 Linked Data

The above scenario, where there is a hierarchy of multiple instances of PCRs, could be made more sustainable by enabling them to use linked data technology to share file format information through a combination of automatic notification and manual administration.

Linked data is about exposing, sharing and connecting related, structured data via the Web and specifying the best practice for doing so. The key technologies involved with linked data are Uniform Resource Identifiers (USRs) to generically identify entities and concepts; Hypertext Transfer Protocol (http) to provide a mechanism for retrieving or describing resources; and Resource Description Framework (RDF), to provide a generic data model to structure and link related data[11].

Developing the PCR in line with linked data principles could lead it to become a system that is better able to interact with semantic technologies. It could provide a system that supports the requirements of the wider digital preservation community, by providing the technological potential to share file format, and other technical data. This could make the development of the PCR more efficient as it will be possible to glean and share technical data from other specified registries and instances, where appropriate. It would also enable the delivery of a database that is easily extensible, and designed without the complexity of standard database schemas and redesigns.

Such development could enable:

- All data that is published in the PCR to be optimized for reuse by semantic web functionality;
- The data held in a global instance of the PCR to partially update other instances within a linked hierarchy of registries;
- The data held in instances of the PCR to be partially updated by other file format registries and externally linked data resources, if applicable and controllable.
- The new version of the PCR to support the requirements of a single, internationally agreed registry, e.g. UDFR.

There is the potential with such development to deliver:

- An Application Programming Interface and associated documentation enabling the PCR to interact with other applications and data sources;
- An automatic mechanism for identifying updates to the master version of the PCR, and highlighting those fields to an administrator to execute;
- A low-level queryable front-end to the PCR data source, and a non-expert user friendly queryable interface;

11 http://linkeddata.org/faq. Retrieved on 11/5/10.

- A governance and administration model for the new version of the PCR, clarifying how the ownership of any authoritative master version should be managed;
- Administration documentation detailing the mechanism behind the PCR database and providing guidance to system administrators wishing to update and append the master instance of the PCR.
- Technical documentation detailing the low-level aspects of the operation of the PCR.
- A re-deployable and easily configurable client server application capable of sitting on servers within any organization.
- A clear licensing statement setting out how users may use data from the PCR for their own purposes.
- A fully documented and published signature conversion algorithm for format identification tool DROID, and other format identification tools, in order to enable them to use generic signatures from the PCR.
- API functionality, to provide a mechanism to output alternative plain byte sequence signature expressions from the PCR database that may be used by identification tools other than DROID.

4. Appendix

The requirements listed below have been taken from a draft Software Requirements Document for PCR3 (PLANETS deliverable PC3 D20 V1.R1.M3.). They are requirements that were envisaged for PCR3 or were enhancements for future development. The numbering of the requirements has been kept for ease of cross reference.

1. Data Requirements

1.1 General Requirements

1.1.1 Families

Label	Requirement
S5.1.2.1	The system must maintain a list of the existing file format families, but mark entity families as a deprecated feature of the system, with GDFR facets being used from PCR version 3 onwards.

1.1.2 Other information

Label	Requirement
S5.1.4.7	In future, the system must maintain a list of media write methods.
S5.1.4.8	In future, the system must maintain a list of media access methods.

1.1.3 PUID Assignment

Label	Requirement
S5.1.5.5	It must be possible for the system to allocate a PUID of the correct type, if one is available, to any new entity that can be assigned a PUID (e.g. file format, software package, process etc.) from within the range of PUID values assigned to the registry instance for entities of that type.
S5.1.5.6	In the future, if the installed registry instance does not have any values left in the range of PUID values for an entity type, or was never allocated any values for that type, then the system will not be able to allocate a PUID to any new entity of that type and the user will not be able to save a new entity of that type and should be informed why.
S5.1.5.8	In the future it must be possible for the system to check whether a user-entered PUID for a new entity (of a type that can be assigned a PUID) is within the range of PUID values assigned to the registry instance for entities of that type. The system should issue an overrideable warning if the PUID is not within the allocated range.

1.2 Core Entities

This section covers the information that is stored for the core entities in the system, which are:

- File Formats
- Software Package
- Hardware
- Character Encoding
- Compression technique
- Storage Media

Core entities suffer from obsolescence and their obsolescence affects our ability to render digital information, which is why information about them is stored in the system.

1.2.1 File Format Requirements

1.2.1.1 External signatures

Label	Requirement
S5.2.1.2.4	The system must be able to store provenance information for external signatures.

1.2.1.2 Software Components

Label	Requirement
S5.2.2.1.1	The system must be able to store the name, version and description of components within software, such as those within JHOVE.
S5.2.2.1.2	The system should allow service pack information for software components to be modelled.

1.2.1.3 Software Package Tools

Label	Requirement
S5.2.2.2.9	In the future, it should be possible to record that a property extraction software package tool **cannot** measure a given instance property of a file format. Note: This is so that it is possible to distinguish between those properties that a tool cannot measure and those properties where it is not known (we have no information) whether a tool can measure them.

1.2.2 Storage Medium Requirements

Label	Requirement
S5.2.6.1	The system must maintain a list of storage media components including the following information: • Write type method (linked to maintained list of write type methods). • Maximum write speed • Write protection mechanisms. • Error correction mechanisms. • Maximum data transfer rates • Data access method employed (lined to maintained list of media access types). • Uncompressed data storage capacity. • Compressed data storage capacity. • Number of data storage sides available. • Number of data storage layers available. • Physical dimensions of the media. • Estimated media longevity. • Coercivity rating (in Oersteds) (if magnetic media). • Recommended storage conditions. • Storage notes. • Handling notes.

1.3 Subsidiary Entity Requirements

The system needs to store a variety of information about people, organisations, documentation, intellectual property rights etc. The need to record information on these entities is to support the core entities that the system is really about. This section covers the information that needs to be held on each of these subsidiary entities.

1.3.1 Processes

A (software) process is an action that can be performed by a software package. Processes can be classified as either 'processes for objects' or 'processes for technical environments', where 'processes for objects' includes both 'processes for components' and 'processes for file formats'.

Label	Requirement
S5.3.5.7	In the future, the content invariance descriptions will be more detailed.

1.3.2 Pathways

This section covers the information held about Pathways, which applies to both File Format Pathways and Technical Environment Pathways.

Label	Requirement
S5.3.6.15	It must be possible to store Testbed results with the pathway step they belong to. Note that the Testbed results will be aggregated results at the 'class' level (not at the level of a specific service instance) and most likely contained in an XML document which conforms to a yet-to-be-created Testbed schema.

1.3.3 External identifiers

Label	Requirement
S5.3.9.3	The system must maintain a list of possible identifier types classified by: • File format identifier types. • Character encoding identifier types. • Compression technique identifier types. • Software package identifier types. • Hardware identifier types. • Storage medium identifier types. • Documentation identifier types. • Intellectual property rights identifier types. • Reference file identifier types.

1.4 System Information

This section contains details of the information recorded by the system in order to support some of the functional requirements

1.4.1 Audit Entries

This section contains details of the information recorded by the system to support the auditing of changes to the data held in the database.

Label	Requirement
S5.4.1.4	For file formats and software packages, the system should record the details of the table data held after the data has been inserted or modified. This set of information then forms the history of an entity, showing what information was stored about it on any given date.
S5.4.1.5	For all other core entities, the system should record the details of the table data held after the data has been inserted or modified. This set of information then forms the history of an entity, showing what information was stored about it on any given date.
S5.4.1.6	In the future, for all other entities, the system should record the details of the table data held after the data has been inserted or modified. This set of information then forms the history of an entity, showing

	what information was stored about it on any given date.
S5.4.1.8	For each audit entry, the system should be able to record the following information: a unique transaction identifier
S5.4.1.9	For each audit entry, the system should be able to record associations to concomitant changes that were made. Note: All concomitant changes will be identifiable by matching the modifier and modification time. There is the possibility that the modification time may drift if a set of related changes (transaction) takes a long time to commit. It may be possible/necessary to generate a unique ID for each transaction and record that against each audit entry. It would then be possible to unambiguously identify concomitant changes.
S5.4.1.10	The system must record the history of the internal signature byte sequences i.e. the details of the byte sequence data (byte sequence, position, big endian flag, offset etc.) held after the data has been inserted or modified, as well as recording when it is deleted. In addition, the system must record the date and time of the change, who made it and any comments made by the person making the change.

1.4.2 Version Control, Rollback and Synchronisation

This set of requirements which builds on the auditing functionality in the system is recorded as potential future enhancements for the system.

Label	Requirement
S5.4.2.1	In the future, the system should allow users with appropriate permissions to rollback changes to the data held in the system to a previous version of a record.
S5.4.2.2	In the future, it should be possible to review the changes that a rollback would produce and only if they are approved will the rollback go ahead.
S5.4.2.3	In the future, the system should enable synchronisation between different instances of the system.
S5.4.2.4	The system should provide an interface to export data to XML to transfer data between instances of the registry.

1.4.3 Technology Watch Alerts

PLATO (the preservation planning PLANETS sub-project) has requested a technology watch function in PCR to identify changes to the data which will potentially impact preservation planning and send out alerts to subscribers when these changes occur.

Label	Requirement
S5.4.3.3	In the future, the specified information to generate alerts for should include: • New software package tools (including entries for new versions of existing software package tools) • New, updated or deleted processes for software packages/software package tools • New, updated or deleted withdrawn details • New file formats (including entries for new versions of existing file formats) • Changes (new/updated/deleted) to file format risk scores • Changes (new/updated/deleted) to file format properties (instance or inherent) • New technical environments • New, updated or deleted Testbed information associated with a pathway step

Label	Requirement
S5.4.3.4	In the future, the specified information to generate alerts for should include new, updated or deleted Testbed information associated with a pathway step.
S5.4.3.6	The system should publish each alert in machine-readable format.

1.4.4 Quarantine Area for Testbed Results Storage

Testbed (the testing and research environment PLANETS sub-project) will be storing aggregated results in PCR. These results will be submitted to a PCR web service and stored in a quarantine area until an administrator can review them.

Label	Requirement
S5.4.4.1	The system must provide a quarantine area for the storage of submitted Testbed results (i.e. soap messages).

2. Administration Requirements

This section details the requirements for the web based administration interface for the system. In addition to providing functionality to maintain the data in the repository, the administration interface also provides functionality to search and browse the data for users with the REPOSITORY_READER role.

2.1 General Requirements

Label	Requirement
S7.1.1	The administration user interface must be web-browser-based.
S7.1.2	The master system should be able to support 5 concurrent users.

2.2 Security

This section details the security requirements for the Administration Interface.

The user requirements, on which these system requirements are based, do not have any explicit security requirements stated. In order to support the Audit functionality, it is necessary that users of the system authenticate themselves before making any changes to the data in the system. Without authentication there is no way to provide the details of the creator/editor of a record, unless the users manually enter that information when they edit information.

As such, the following requirements are included to provide a basic level of security and also to support the generation of audit entries when the contents of the database are modified.

Label	Requirement
S7.2.4	The system should automatically log out an inactive user after a set time period.
S7.2.5	The system should allow configuration of the period of inactivity allowed before a user is automatically logged out. Note: configuration will not be done through the GUI.
S7.2.6	It should be possible to configure the system to allow different authentication authorities (e.g. OpenLdap, Novell eDirectory, Microsoft ActiveServer, Database Source) to be used
S7.2.9	In the future, the system should have a separate level of authorisation which will allow users to approve Testbed results, and view the rest of the data, but not edit it.

2.3 Repository Reader Functionality

This section covers the functionality provided to a user with the role REPOSITORY_READER.

2.3.1 Core Entities

This section covers the functionality available to a user with role REPOSITORY_READER which relate to Core Entities.

2.3.1.1 Common

Label	Requirement
	Common – View
S7.3.1.1.14	In the future, the list of related core entities in the detailed view of an individual core entity must be grouped by the type of the related core entity.

2.3.1.2 File Formats

Label	Requirement
S7.3.1.2.13	The detailed view of an individual file format should also include the pathways for which the file format is either the source or target file format.

2.3.1.3 Storage Media

Note: the storage of detailed storage media data is an enhancement.

Label	Requirement
S7.3.1.7.1	In the future, the system must allow users to list all storage media in such a way as to meet all the common requirements for listing core entities.
S7.3.1.7.2	The system must allow users to view an individual storage medium in such a way as to meet all the common requirements for viewing an individual core entity.

2.4 Repository Administrator Functionality

This section details the requirements for a user with the role REPOSITORY_ADMINISTRATOR (described as 'an administrator' in the requirements in this section).

Users with the REPOSITORY_ADMINISTRATOR role can carry out all functionality available to a REPOSITORY_READER.

2.4.1 Audit Trail

This section details the requirements for the display of audit information.

Label	Requirement
S7.4.3.3	The detailed view of an individual audit entry should include the following information: • modification time • modification type • name/identifier of the person who modified the record the state of the record after each auditable event
S7.4.3.4	When users make a change (create, modify or delete) to an internal signature byte sequence the interface must allow (but not force) them to record any comments they have on the change they have made.
S7.4.3.5	An administrator should be able to view the audit entries for any audited entity in the registry.
S7.4.3.6	The detailed view of an individual audit entry should include the following information: • the administrator that performed the action • the information that was added/modified • the date and time that the modification took place • the type of modification (add/modify/delete)
S7.4.3.7	In the future, the detailed view of an individual audit entry must include the transaction identifier that is used to identify concomitant changes.

Label	Requirement
S7.4.3.8	In the future, when displaying the audit information for changes to internal signatures, the extra audit information recorded for internal signatures (as defined in S5.4.1.10) must be displayed.

2.4.2 Lists of Values

This section details the requirements for maintenance of the lists of values used to restrict selections for particular fields in the system.

Within the system, the following are considered to be lists of values (LOV):

Agent Relationship Types	External Identifier Types
Agent Roles	External Signature Types
Agent Types	Flag States
Byte Sequence Positions	Format Byte Orders
Component Types	Format Disclosure Levels
Compression Lossiness Types	Intellectual Property Right Jurisdictions
Content Variance Types	Intellectual Property Right Types
Countries	Languages
Document Content Types	Pathway Roles
Document Types	Pathway States
Documentation Availabilities	Pathway Types
Entity Class Ext Id Types	Process Action Types
Entity Classes	Process Types
Entity Families	Programming Languages
Entity Relationship Types	PUID Types
Entity Types	Software Package Interface Types

They are characterised by the following:

- They are populated with values as part of the system installation.
- They are used to populate drop down lists of values.
- They are infrequently modified.
- They may be used to determine behaviour of the system in workflows.

Label	Requirement
S7.4.4.2	The system should provide an interface to list all entries for each of the LOV tables.
S7.4.4.3	A refresh button should be added to the main screens to update drop-down lists.
S7.4.4.4	The system should provide an interface to view individual entries in each of the LOV tables.
S7.4.4.5	It should be possible for administrators to maintain all authority-controlled lists of values.
S7.4.4.6	When viewing a list of values, it should be possible for an administrator to delete an individual entry from that list, unless that entry has been assigned to an entity in the system.
S7.4.4.7	It should be possible for administrators to edit an individual list of values entry.
S7.4.4.8	When viewing or editing an individual list of values entry, it should be possible for an administrator to delete that list of values entry.
S7.4.4.9	When editing an individual list of values entry, the system should indicate that it is in edit mode.
S7.4.4.10	When editing an existing list of values entry, it should be possible for an administrator to add or remove items to/from the displayed lists of associated entities.

Label	Requirement
S7.4.4.11	When editing an existing list of values entry, it should be possible to save all changes to the individual list of values entry.
S7.4.4.12	When editing an existing list of values entry, it should be possible to abandon all changes to the individual list of values entry (e.g. by navigating away from the page without saving the changes).

2.4.3 Review Testbed Results

This section covers the requirements relating to enabling an administrator to review submitted Testbed results and either accept or reject them.

Label	Requirement
S7.4.7.1	It must be possible for an administrator who has the authority to approve results from a given instance of Testbed to review results submitted from that instance to the system's quarantine area.
S7.4.7.2	It must be possible for an administrator to select which pathway step to associate the Testbed results with.
S7.4.7.3	Approval of the results by the reviewer must result in them being associated with the selected pathway step.
S7.4.7.4	Rejection of the results by the reviewer must result in the results being deleted from the system's quarantine area.
S7.4.7.5	Inserting new Testbed results, or updating existing Testbed results belonging to a pathway step, will overwrite the existing data in their entirety.

2.4.4 PUID Range Assignment

This section covers requirements relating to PUID assignment.

Label	Requirement
S7.4.8.1	In the future, it must be possible for an administrator to view and update the list of PUID number ranges for the installed registry instance for each type of entity that can be assigned a PUID (e.g. file format, software package, process etc.).

3. Interfaces for External Systems

This section covers the interface requirements that are not browser based. This covers web services based on SOAP or REST.

3.1 REST Interface

This section details the Representational State Transfer (REST) interface that the system must provide.

Label	Requirement
S8.1.1	The Core Registry should expose its contents via a REST interface as a result of the following queries: • Get Identify Tools • Get Validate Tools for PUID • Get Property Extract Tool for PUID • Get Object Extraction Tool for PUID • Get Migrate pathway for Start PUID and End PUID • Get Format Risk for PUID • Get Redaction Tool for PUID

3.2 SOAP Web Services

This section details the SOAP web services that the system must provide.

3.2.1 Characterisation

This section details the SOAP web services that the system must provide as part of the system's support for characterisation.

Label	Requirement
S8.2.1.1	For backward compatibility with Pronom 6.2, the system should implement the following web services • getIdentificationTools • getValidationTools • getPropertyExtractionTools • getToolProperties • getObjectExtractionTools • getSchema • getDTD in accordance with the wsdl found at http://www.nationalarchives.gov.uk/PRONOM/Services/Contract/PRONOMcharacterisation.wsdl.

3.2.2 Preservation Planning

This section details the SOAP web services that the system must provide as part of the system's support for preservation planning.

Label	Requirement
S8.2.2.1	The system must, for backward compatibility with Pronom 6.2, implement the web services below in accordance with the wsdl with the namespace of http://pp.pronom.nationalarchives.gov.uk: • getFormatsAtRisk • getFormatsByMigrationType • getFormatRisk • getFormatPropertyRisk • getMigrationPathways • getComponentIdentificationTools • getComponentMeasurementTools • getComponentProperties • setPathwayApproval • setPathwayCurrent

3.2.2.1 Preservation Planning – File Formats

This section details the SOAP web services that the system must provide related to preservation planning and file formats.

Label	Requirement
S8.2.2.1.2	The system must implement a web service called getInherentProperties that returns all inherent properties. This webservice does not need any inputs.
S8.2.2.1.3	The system must implement a web service called getInstanceProperties that returns all instance properties belonging to a single file format as identified by its PUID. Without input it will return all instance properties in the system.

Label	Requirement
S8.2.2.1.5	The system must provide an interface to retrieve a list of risk scores, from both inherent and instance properties, for a specific file format as identified by its PUID. Note: This would allow, for example, Preservation Planning tools to determine the particular properties that apply to a specific format and which can be used to populate nodes of a decision tree. This requirement has been requested by PLANETS sub-project PP/4.
S8.2.2.1.7	The system should provide an interface to retrieve an XCDL schema file for a specific file format.

3.2.2.2 Preservation Planning – Migration Pathways

This section details the SOAP web services that the system must provide related to preservation planning and Migration Pathways.

Label	Requirement
S8.2.2.3.2	The system must, for backward compatibility with Pronom 6.2, implement the web services below in accordance with the wsdl with the namespace of http://pp.pronom.nationalarchives.gov.uk: • getMigrationPathways

3.2.3 Other Web Services

Label	Requirement
S8.2.6.1	The Core Registry should be able to select all Technical environments that are capable of supporting a series of format PUIDs.

3.2.4 Testbed Results

This section details the SOAP web services that the system must provide as part of the system's support for storing and retrieving aggregated Testbed results.

Label	Requirement
S8.2.7.1	The system must implement a web service (submitTestBedResult) to allow Testbed to submit aggregated results to the system. This web service will accept the Testbed results (as a CLOB probably containing an XML document) together with enough information to uniquely identify the pathway step the results should be associated with.
S8.2.7.2	The system must implement a web service to allow the Testbed results associated with a given pathway step to be retrieved. The web service (getTestBedResults) will take the pathway PUID and step sequence number to uniquely identify the pathway step, and will return the Testbed results (as a CLOB probably containing an XML document).
S8.2.7.3	The system must implement a web service (findPathwaySteps) to allow a user to determine which pathway step they want the results from. The inputs are a software package PUID, file format source PUID, file format target PUID and an optional parameter string. The response contains information on all pathway steps which match this data (0 or more). As well as descriptive data, it returns the pathway PUID and step sequence number required to call the previous web service.

3.3 OAI-PMH Interface

This section details the Open Archives Initiative – Protocol for Metadata Harvesting (OAI-PMH) service interface that the system should provide.

Label	Requirement
S8.3.1	The Core Registry should expose its contents via an OAI-PMH interface for harvesting metadata.

4. System Wide Requirements

This section covers the requirements of the system needed to ensure that the product can actually be delivered and used and to ensure long-term quality.

4.1 Platform Requirements

The system will use two different web servers:

- The public web interface (see section 0) and REST interface (see section 3.1) will use Microsoft's IIS web server.
- The administration interface (see section 2), SOAP web services (see section 3.2), and OAI-PMH interface (see section 3.3) will use a web server which supports version 6 of Java Virtual Machines (JVM).

Label	Requirement
S9.1.7	The system should use Tomcat 6 to provide the OAI-PMH Interface.

4.2 Quality, Reliability and Maintainability Requirements

Label	Requirement
S9.5.1	The system should make use of XSL transformations in order to avoid hard-coding report structures.
S9.5.4	All HTML should be XHTML1.0 Transitional and CSS 2 compliant.