




Research data sets executable workflows for large-scale execution

Authors

Alastair Duncan, Catherine Jones, Antony Wilson (Science and Technology Facilities Council), William Palmer (British Library), Stefan Proell (University of Technology Berlin)

May 2014

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

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

Executive Summary

The main purpose of this deliverable is to report on the work which has been undertaken for the 5th period of WP17 - TB.WP.3 which is part of the SCAPE project. The main bulk of the work was undertaken by STFC with contributions from BL and TUW.

STFC created refined workflows that enabled the parallelisation of workflows described in D17.1 in order to provide a scalable preservation environment for research datasets. The workflow used 3rd party software within the workflow as it is not practical to recode and maintain the software to run on the default SCAPE parallel platform, Hadoop. This proved problematical as some of the 3rd party software uses specialised file formats and high performance access methods that are not compatible with the Hadoop filing system and HDInsight, a Hadoop variant. A tool which allows the wrapping of 3rd party applications and scripts so that they can be executed on a Hadoop cluster was used within the workflow. This worked for a small data set but issues were encountered for larger data sets. These issues have been documented here.

STFC's work also involves the building and maintenance of a Research Object with specific specialisations for STFC Research Data and is focused specifically at the Investigation. This brings all of the relevant objects together so that the Research Object as a whole can be understood. Any data or information that is relevant to an investigation that enhances the understanding of the investigation can be added. The specific challenge is to add enhanced information and understanding to the Research Object with a more automated process reducing the labour intensive nature of the process. This is attempted by starting with standard metadata from the Core Scientific Metadata Model and additional information is added from the Photon and Neutron data infrastructure Knowledge Organisation System ontology in addition to the raw experimental data, metadata, software, derived data etc. Inferences are made by matching the techniques which are applied by specific target stations to the data that has been produced at that target. This is stored in a Resource Description Framework triple store which is queried with SPARQL. This is then archived using tools developed within the SCAPE project to a repository. Further work is planned to monitor the state of the object and update it based on the expiry of the embargo date.

BL has done work on identification, validation and checksumming of a complex corpus. A 1.4TB corpus of UK Ordnance Survey geospatial data has been deposited with the British Library for access and long-term preservation and cannot be shared. There are a variety of file types within the data set and BL are attempting to understand the data by generating a checksum for each file and using characterisation techniques to obtain information that can be used to generate a signature which can then be used to identify the file type. BL have also created a validation tool which can be used on the 2 most common file formats in the data set. The software is currently called GeoLint. This work is still at an early stage.

BL's work also involves the Normalisation of Disparate Tabular Data Sources. BL has a data set which comprises a number of different file formats. The data has been deposited by local authorities within Britain and is private. The work was to normalise these into a common format so that any file analysis or processing can be done in a common way. The normalisation work has been halted mainly



because the data set is not very large and unlikely to grow larger and there was no other data available. The normalisation software written by BL which runs on Hadoop has been made publically available.

TUW's work aims to persistently identify data sets from the metadata that was used to generate the data set in order that the data set can be accurately regenerated without having to store a copy of the data set which could be large. The work is quite abstract currently as it is at an early stage of development.

Table of Contents

Executive Summary	iii
1 Introduction	1
2 STFC Contribution	2
2.1 Context	2
2.2 Migration from local format to domain standard format (user story)	3
2.2.1 Parallel platform	4
2.2.2 Distribution of software	5
2.2.3 ToMaR (Tool-to-MapReduce)	5
2.2.4 Approach to workflow implementation	6
2.2.5 Taverna workflow for the file format migration using ToMaR (Linux)	6
2.2.6 Discussion	7
2.3 Preserving the context and links to research data or preserving research objects	10
2.3.1 Approach to implementation	10
2.3.2 Results	13
3 BL Contribution	13
3.1 Normalise Disparate Tabular Data Sources	13
3.2 Identification, validation and checksumming of a complex corpus	14
4 TUW Contribution	15
4.1 Persistent Data Citation of Dynamically Created Subsets	15
4.1.1 Approach to implementation	15
5 General Conclusions	16
6 Glossary	18

1 Introduction

The Research Data Testbed is concerned with research data as a material type. This term covers a wide range of domains, preservation issues and file types, so the work within this work package is aiming to address examples of workflows which can be generalised to the wider community.

Contributors to the WP17 - TB.WP.3 work package are STFC (36 months effort), BL (9 months effort) and TUW (3 months effort). The work effort for each organisation varied considerably and this is reflected in the size of sections within this document.

The two content holders, BL and STFC, demonstrate the difference between those whose remit is to collect and preserve content nationally, and thus have little say over the formats and standards used, and an organisation that is responsible for the generation, use and preservation of the data.

Both of the BL uses cases are concerned with activities to ensure that data which has been created elsewhere and deposited for long-term preservation with the BL can be managed for the long term. The first use case aimed to normalise deposited material which comes from a large number of geographic disparate organisations although the intellectual content is of a fixed specification. By normalising the content into a standard layout, the intellectual meaning is not changed but the management overhead is reduced. The second BL use case concerns geospatial information and the activities around this data are to ensure that the BL knows what the data is and whether there have been any changes to it. As a deposit organisation, understanding what has been deposited and managing this effectively is very important.

STFC's first use case is one of format migration, from a local format into a domain standard that they have been involved with specifying, this is migration to ensure that the sustainability of the content within a format is maintained. By moving from a local, only understood and produced at ISIS, to one which is in use in the Photon and Neutron community, there is the possibility to reduce the dependence on local knowledge for long-term storage. This is especially relevant as the facility is now 30 years old and the youngest of those who worked on the original instruments are now retiring. The second case study from STFC is investigating what additional context needs to be preserved to maintain the usefulness of the original object over time, this is not a problem solely related to research data, but is of particular importance as the contents of research data files are particularly difficult to comprehend as they do not usually contain natural language or images but need to be opened and read by specialist programmes.

The work that TUW describes is addressing the issue of how to cite and preserve data sets generated as a result of database queries.

This document describes the different use cases and research contexts, then discusses the solutions developed to address the needs, and finally outlines the lessons learnt as a result of the experimentation.

2 STFC Contribution

2.1 Context

The data sets provided by STFC¹ to the SCAPE project are generated by the ISIS² neutron spallation source. This scientific research facility is one of a small number of worldwide facilities that use either neutrons or photons to explore the structure of matter, behaving as very high powered microscope. The researchers who use this and the other facilities worldwide come from a very diverse set of domain communities, from those researching the timbers on the Tudor warship the Mary Rose to those looking at reactions on protein chains; what binds them together is the use of the instrument.

In essence in experiments undertaken on these facilities, a sample of the material is placed within the instrument and a beam of high energy neutrons, muons or photons is aimed at the target sample and the detectors then collect information on the resulting scattering or diffraction patterns. Using models and analysis, these detection patterns can be used to generate the structure of the sample. Each of the experiments undertaken at ISIS is done by different Principle Investigators looking at different samples, so that over 2000 different experiments are done in a usual year.

As these facilities are expensive to build and run it is usual for them to operate as a national or international facility. They are also custom-made, the ISIS facility first started in 1984 and thus used custom made detectors and local standards for the data files.

Standardisation moves within the community, especially driven in Europe by the PaN-data³ consortium of 13 photon and neutron sources has created the domain standard NeXus⁴ Format for the file format and it is also building a common data management infrastructure to provide common experience for users of more than one facility. Currently ISIS holds data in both formats and there are migration routines available to transfer from the local format to this new domain format.

As each facility is responsible for the computing infrastructure, there is tight control over the process from proposal to data storage, at this point the researcher will take the data away and perform further analysis to enable conclusions to be drawn and published. Information on the publications resulting from experiments is important to facilities as it provides metrics on impact and effectiveness.

One of the areas becoming more important as the policy environment for data management is encouraging the use, re-use and preservation of publicly funded data, is the consideration of what additional contextual data is required to enable the data to remain meaningful.

The experiments and workflows generated by STFC using ISIS data can be considered to be addressing a very specific scientific domain however the general principles apply regardless of domain. In the scenario of format migration, the stages of ascertaining the type of material, performing a translation function and quality assurance are standard and some of the issues which have occurred are due to size and set-up constraints not the type of file themselves and so have wider applicability. The work on contextual information applicable to all data being preserved, but

¹ <http://www.stfc.ac.uk>

² <http://www.isis.stfc.ac.uk>

³ <http://pan-data.eu/>

⁴ <http://www.nexusformat.org/>

this applicability depends strongly on context. The work done by STFC, therefore, can help to identify issues and mechanisms for solving issues which are of interest to those beyond the neutron and photon community.

2.2 Migration from local format to domain standard format (user story)⁵

This section describes the processes and tools used to migrate raw data and associated metadata generated from ISIS instruments to a domain standard format. It also discusses some of the problems that have been encountered using the Hadoop parallel platform.

Data files captured from the instruments are typically captured in raw format along with a number of metadata files which capture information such as runtime, instrument settings, events, logging etc. Depending on the age of the instrument which is capturing the data the file format will either be raw or Nexus, and in some cases both file formats are concurrently produced. The Mantid⁶ analysis software has been written to be able to analyse both raw & NeXus formats by incorporating a conversion routine within the software suite. This doesn't mean that the original file is permanently migrated as a preservation action, just that the content of the file is converted to enable scientific analysis. However, there is a large quantity of historical data which is still in its raw format and needs to be migrated to the NeXus format as a preservation action. Once the raw files have been converted there is a need to check the file for validity and this is done by first checking the file type and then by checking that the metadata contained within the NeXus format file is valid. Once it has been ascertained that the file is valid a checksum has to be generated and this can be used to detect any changes over time in the file due to errors with storage, transmission or tampering.

NeXus format files are the preferred format for storage of data, metadata and access to the data for analysis. There are tools that have been developed for fast and efficient access to NeXus format files for various languages including C, C++ and Java. The Mantid project has developed software to read raw files and write to NeXus format with plugins for analysis of data which are stored in the file. The Mantid software used in the migration is raw2nexus. NeXus format is a common data format for neutron, x-ray and muon science. The definitions lay out the data in a standardised way. It has been developed as an international standard by scientists and programmers representing major scientific facilities in Europe, Asia, Australia, and North America in order to facilitate greater cooperation in the analysis and visualization of neutron, x-ray, and muon data. A NeXus file can be represented in 3 formats HDF5⁷, HDF4 and XML. HDF5 is the format recommended because of its efficiency and speed although HDF4 is supported and XML should only be used for small data sets due to its poor performance.

HDF5⁸ is a data model, software library, and file format for storing and managing data. It supports an unlimited variety of data types, and is designed for flexible and efficient I/O and for high volume and complex data. HDF5 is portable and is extensible, allowing applications to evolve in their use of HDF5. The HDF5 Technology suite includes tools and applications for managing, manipulating, viewing, and analysing data in the HDF5 format.

⁵ <http://wiki.opf-labs.org/display/SP/Migration+from+local+format+to+domain+standard+format>

⁶ http://www.mantidproject.org/main_Page

⁷ <http://www.hdfgroup.org/>

⁸ <http://www.hdfgroup.org/HDF5/>

Four data sets were compiled for use in this User Story:

- Small data set 1.2Gb 33 raw files with metadata files ranging in size from 24Kb to 129Mb
- Large data set 1.1Tb 24965 raw files with metadata files ranging in size from 17Kb to 445Mb
- Large data set made up from copied files with metadata files from the Small Data set 1.2Tb 33000 raw files ranging in size from 24Kb to 129Mb
- Large data set made up from copied files with metadata files from the Small Data set 1.1Tb 147687 raw files with metadata files ranging in size from 24Kb to 265Kb

2.2.1 Parallel platform

The default parallel platform used in the SCAPE project is Hadoop⁹. The Apache Hadoop software library is a framework that allows for the distributed processing of large data sets across clusters of computers using simple programming models. It is designed to scale up from single servers to thousands of machines, each offering local computation and storage. Rather than rely on hardware to deliver high-availability, the library itself is designed to detect and handle failures at the application layer, so delivering a highly-available service on top of a cluster of computers, each of which may be prone to failures. Hadoop uses the Hadoop Distributed File System (HDFS™) which is a distributed file system that provides high-throughput access to application data.

2.2.1.1 Information on STFC's local cluster

The Hadoop cluster used so far in SCAPE at STFC is maintained and provided by other members of the STFC Scientific Computing Department and is a test facility only available within the STFC firewall. The Hadoop cluster has six slaves providing 70TB of HDFS storage and 24 MapReduce slots; these are managed by two virtual machine head-nodes, the HDFS NameNode and MapReduce Job Tracker. Each slave machine has both a map reduce task tracker and HDFS DN to enable the minimum movement of data to the compute resource. The OS used is Scientific Linux which is a scientifically enhanced version of Red Hat Enterprise Linux.

2.2.1.2 Azure¹⁰ and HDInsight¹¹

Azure is a cloud based service that is run by Microsoft. It has facilities for running Windows and Linux virtual machines and has mechanisms for provisioning high performance computing (HPC)¹² clusters and Hadoop clusters (a Hadoop cluster is a HPC cluster specialisation that uses the MapReduce¹³ programming model) that run on the MS Windows platform as well as the more standard Linux and Windows virtual machines. The Hadoop implementation is from Horton Works and called HDInsight. The distributed storage systems on Azure are Blob, Table and Queue. The storage system used with HDInsight is Blob. It has the scheme type of `wasb://` and differs significantly from `hdfs://`

⁹ <http://hadoop.apache.org/>

¹⁰ <http://azure.microsoft.com/en-us/>

¹¹ <http://azure.microsoft.com/en-us/services/hdinsight/>

¹² <http://insidehpc.com/hpc-basic-training/what-is-hpc/>

¹³ <http://research.google.com/archive/mapreduce.html>

There is an HDInsight emulator which can be installed on a Windows machine. This works well but uses HDFS emulation. Code which runs on this will not necessarily run on an HDInsight cluster running in the Azure cloud as HDFS and WASB have significant differences which are discussed later.

Azure has an HDInsight client application that, once installed, can be run from the Windows desktop and this is used to submit a job to an HDInsight cluster running in the Azure cloud. This is a different way of submitting jobs to a Linux based Hadoop system which requires the user to be logged in to a node in order to submit a job. A Taverna workflow will need to be created that utilises the HDInsight client job submission API which is the windows equivalent of the ToMaR workflow developed for the Cloudera version of Hadoop (shown later).

2.2.2 Distribution of software

The Hadoop clusters being used by the various testbeds within SCAPE are using a Linux based operating system. At STFC this is a Redhat¹⁴ derivative Scientific Linux 6¹⁵, on other testbeds it is the Debian¹⁶ derivative Ubuntu¹⁷. In most cases the users have access to the cluster to install software. This can be using standard distribution packages, by custom packages from distributors such as <http://rpm.pbone.net/>, or by installing the software on each node in user space. All of these require access to the node. HDInsight does not allow access at all to the node to install software. There are 2 mechanisms that can be used to distribute the software:

- It can be distributed within the jar when submitting the job. If this is not a java application but a 3rd party binary this requires the programmer to unpack the software and put it somewhere where it can be executed.
- The alternative is to pass it with the job using the `-files` option and let the Hadoop system unpack it onto the node in the temp space. If using the ToMaR wrapper the ToolSpec will have to reflect where Hadoop will place the binary. This should be straightforward with the relative file construct `./` as the binary should then be executed from the current working directory (where it has been copied to).

2.2.3 ToMaR (Tool-to-MapReduce)¹⁸

ToMaR is a tool developed within the SCAPE project to allow 3rd party software tools to be executed on a Hadoop cluster. ToMaR wraps the software which is described by SCAPE Tool Specification Language for distributed execution in a MapReduce job. Hadoop requires a custom mapper and reducer (optional, not used in STFC work) method to be written which will be executed on a cluster node. ToMaR provides the mapper and takes care of copying files required by the 3rd party tool to the local node file system where the tool will be executed with the required files as inputs. ToMaR is used in all stages of the raw2nexus migration. A list of files to be migrated and checked is generated from files that are located in HDFS. This is passed to ToMaR along with the specification of the tool which is to be executed. ToMaR will copy the files, which will be used as inputs to the tool, to a local directory, execute the tool and then copy any outputs from the tool back to the Hadoop file system (generally HDFS but it could be WASB (Windows Azure Storage Blob))¹⁹

¹⁴ <http://gb.redhat.com/>

¹⁵ <https://www.scientificlinux.org/>

¹⁶ <http://www.debian.org/>

¹⁷ <http://www.ubuntu.com/>

¹⁸ <https://github.com/openplanets/tomar>

¹⁹ <http://azure.microsoft.com/en-us/documentation/services/storage/>

2.2.4 Approach to workflow implementation

The 5 stages of work flow and the software used in each stage:

1. *raw2nexus conversion*

The raw2nexus conversion takes the raw files and metadata files and puts them in to a HDF5 file based on the NeXus Format. The NeXus format specification, tools and libraries are mature software, however, there are a number of dependencies and this can prove problematical when distributing the software on a cluster. The raw2nexus software was compiled statically to avoid any possible conflicts with software packages already installed on cluster nodes.

2. *Filetype checking software*

Format Identification for Digital Objects (FIDO)²⁰ is a Python command-line tool to identify the file formats of digital objects. It is designed for simple integration into automated work-flows. FIDO uses PRONOM signatures to try and determine the file format. The FIDO tool is used for checking the file type of the generated NeXus format file.

3. *extraction of metadata in xml format*

One tool that is bundled with the NeXus format tools is the nxconvert utility. This will convert to and from HDF5, HDF4 and XML. There are 2 XML formats, metadata and raw data or just metadata. The extraction in this case is just the metadata. This can then be validated against an XML schema or a Schematron²¹ file generated from a Nexus Definition Language (nxdl)²² file.

4. *validation of xml metadata*

A further tool bundled with the NeXus format tools is the nxvalidate utility. This takes a nxdl file and generates a Schematron file from this. Schematron is a language for making assertions about the presence or absence of patterns in XML documents. This generation is an overhead which can be done once and then the resulting Schematron file can be used to validate all of the generated NeXus format files. This proves to be an efficient approach to validation of the NeXus format files. Probatron²³ is a family of free, open-source tools that implements the Schematron language, allowing the checking of XML content using ISO Schematron schemas and this was used with the generated Schematron file to validate the XML metadata file.

5. *checksum generation*

A simple python checksum generator was used to generate a md5 checksum from a stream. This enables the creation of a checksum from very large files without having to hold the complete file in memory.

2.2.5 Taverna²⁴ workflow for the file format migration using ToMaR (Linux)

Taverna is an open source and domain-independent Workflow Management System which consists of a suite of tools used to design and execute scientific workflows. It was used to orchestrate the submission of jobs to Hadoop for migration, validation and checksum generation. The blue boxes in the diagram below are inputs and outputs. The orange boxes are the ToMaR jobs.

²⁰ <https://github.com/openplanets/fido>

²¹ <http://www.schematron.com/>

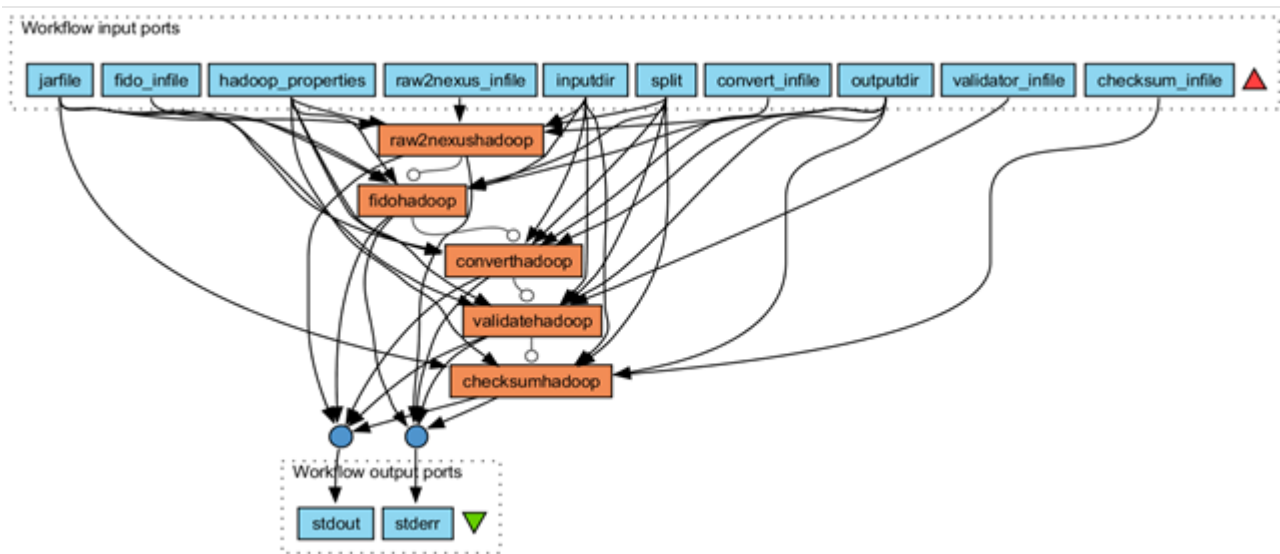
²² <http://download.nexusformat.org/sphinx/nxdl.html>

²³ <http://www.probatron.org/>

²⁴ <http://www.taverna.org.uk/>

Inputs:

Input name	Description
jarfile	this is the java code for the ToMaR software
xxx_infile	these are the files which list the software to be executed by ToMaR and the files to be worked on
split	The number of lines the infile is to be split into per Hadoop task
hadoop_properties	a file with configuration properties for the Hadoop job
input_dir	the directory where the Toolspec files are located
output_dir	the directory where outputs are written to



The ToMaR workflow developed for the Cloudera described above has been tested on both the Cloudera Hadoop platform and on the HDInsight emulator. This ran through to completion using the small data set on both systems but on the emulator it ran slowly. It's not possible to run the workflow as is in the cloud base HDInsight as the job submission mechanism is different and ToMaR has to be adapted to take into consideration the file system differences.

2.2.6 Discussion

A number of problems occurred when running large scale jobs on the Hadoop cluster at STFC. The small data set was first used to test the workflow and configuration options for the job and this worked well. The job completed, the NeXus files were generated and validated and a checksum for the generated NeXus file was created. When a larger data set was used by the workflow, the Hadoop system memory started to swap after about an hour and once that started the system slowed to a crawl and eventually the job had to be killed and the Hadoop system restarted after a disc space clean up. A reconfiguration of the job so that the number of tasks that were run on a node was restricted to one at a time improved the performance but the Hadoop system started to swap at between 11 - 16hrs into the job and again the job could not complete. When the data set of smaller files was used the Hadoop system lasted longer before it failed, the larger the file size the shorter the time to failure. Hadoop has been designed to run on low cost equipment with data being split in to small chunks and processed in parallel in the mapper tasks with the reducer tasks combining the results. The major problem that has not been overcome in this work package is what happens when the file size is large and it cannot be split into small enough chunks that can be handled on the nodes

in the cluster. In the case of the STFC story, the file can't be split in to chunks at all as the HDF5 library is not compatible with the HDFS file system. Copying small HDF5 files from HDFS to the local file system worked reasonably well but problems occurred with larger files. Much more disk space and memory is required, particularly if more than one task is to be run on a node concurrently. This is not compatible with one of the principles of Hadoop, namely that it can run on low cost hardware.

2.2.6.1 Number of tasks per node

Hadoop nodes can be configured to run a specific number of tasks per node. For example, if a node has 8 cores then it could be assumed that up to 8 tasks could be run in parallel per node. This is in addition to the parallelisation of the job by the number of nodes that are available. When the Hadoop system at STFC was configured like this it did not work well. ToMaR has to copy the files to be worked on from HDFS to the local system and then execute the tool over the files and copy back the modified files. HDFS takes care of making the files available on the appropriate nodes but there is still significant input/output in copying the file to a different location locally. The size of files in the test data set range from small, in KBs up to large 100s MBs. A NeXus format file is roughly 3 times the size of the original raw file. It was found that the jobs failed with disc space errors as there was not enough disc space allocated to the local file system to allow for this, particularly when a number of tasks were running in parallel using all of the available cores. A reconfiguration of the Hadoop job that ensured that only one task was run at any one time on a node improved performance. The job ran for much longer before the Hadoop system ran out of resources.

2.2.6.2 Custom mapper

There was an option of writing a custom mapper so that Hadoop would do the migration natively rather than using 3rd party binaries (raw2nexus, NeXus tools and HDF5). This would require the use of the NetCDF²⁵ library or SciHadoop²⁶ (has a dependency on NetCDF) that could read and write to HDF5 files that are located on HDFS. This would have entailed rewriting the NeXus tools library and the raw2nexus migration tool (has a dependency on NeXus tools library). This was considered out of scope of the project.

2.2.6.3 Distribution of software

In practice the files do not seem to be distributed via the distributed cache, at least not to where they are expected. The working directory was listed during execution and this showed the files that had been copied correctly by ToMaR but no files which should have been copied by Hadoop. This may be a good extension for ToMaR to consider, bundling the 3rdParty tool and dependencies extracting and setting execution permissions and library paths. The tools can then be put onto HDFS/WASB and would be accessible from any of the nodes.

A printout of the current relative path of the working directory is:

```
c:\apps\temp\hdfs\mapred\local\taskTracker\admin\jobcache\job_201404100830_0002\attempt_201404100830_0002_m_000000_0\work
.INTER00011372.raw.crc
.INTER00011372_ICPdebug.txt.crc
.INTER00011372_ICPevent.txt.crc
.INTER00011372_ICPstatus.txt.crc
.INTER00011372_Status.txt.crc
INTER00011372.raw
```

²⁵ <http://www.unidata.ucar.edu/software/netcdf/>

²⁶ <https://github.com/four2five/SciHadoop>



```
INTER00011372_ICPdebug.txt  
INTER00011372_ICPevent.txt  
INTER00011372_ICPstatus.txt  
INTER00011372_Status.txt
```

This shows the files copied by ToMaR but no software distributed.

2.2.6.4 Windows Storage Azure Blob (WASB)

Blob storage does not have a directory structure; however, it does have constructs such as:

```
/user/ad43/myfile.txt
```

It may look like a directory structure but what is known on most file systems as a separator is just a textual element in a file name on WASB. This can cause problems as Hadoop has the concept of directories and tools to test whether a path is a directory and testing whether a path is a directory will always return false or cause an IOException e.g.

```
Path p = new Path("/user/ad43/myfile.txt" );  
boolean isDir = p.getFileSystem().getFileStatus(p).isDirectory();
```

`isDir` will be false in the case above

```
Path p = new Path("/user/ad43" );  
boolean isDir = p.getFileSystem().getFileStatus(p).isDirectory();
```

In this case an exception will be thrown because the directory `/user/ad43` does not exist. However, if there is a blob in the file system with the name:

```
/user/ad43
```

and the same code is executed `isDir` will be false as it is not a directory but a blob. A further gotcha is that `/user/ad43` and `/user/ad43/` are 2 different blobs as is `user/ad43` and `/user/ad43/` whereas, in many file systems this would refer to the same directory.

A further quirk is that with the WASB file system there is the concept of a container, which is effectively a name space and this is considered as a separate file system in HDInsight. This has implications for differing file systems within a mapper as data will be stored in a different container to the one HDInsight will be using as its default filing system. This is because when an HDInsight cluster is created, a container is created and all files required for the running of the system are created in this container. The programmer has to be conscious of this when writing code as it cannot be assumed that the data is located in the default filing system, in fact it will never be there by default, it would have to be copied across after the cluster has been created and each time the cluster is created. A better strategy is to create different `FileSystem` objects which correspond to the containers being accessed from the cluster.

The model of usage with Azure and HDInsight is that the service is started where upon the customer is then being charged and when the job or service is not required any more it is torn down. The customer is then only charged for the time the cluster was in use. The storage container that was being use for the cluster is not destroyed but it is not reconnected when the cluster is restarted (the cluster is not in fact restarted it is created and deleted, this is different from an Azure HPC cluster

which is created and then started and stopped) and so any context between runs is lost. It can be referred to if the container is specified in the specific paths being used but this would have to be passed in with the job as configuration parameters e.g.

```
wasb://<container_name>@<storage_account_name>.blob.core.windows.net  
/user/ad43/raw2nexusinputSmallInsight.txt
```

Whereas this command would use the default system (the container which is created when the cluster is provisioned):

```
wasb:///user/ad43/raw2nexusinputSmallInsight.txt
```

2.3 Preserving the context and links to research data or preserving research objects

One of the key preservation issues is ensuring that the meaning of the digital object is also kept for the long-term alongside the object. Being able to open an object doesn't necessarily mean that the contents can be understood. This issue also applies to physical objects, an example of this are the Bronze Age Hittite Cuneiform²⁷ tablets on them discovered in Turkey, the meaning of the texts/writing was only deciphered in the 20th Century. To address this issue STFC have been working on building and preserving Investigation Research Objects.

The aim of Research Objects²⁸, and the special instance of an Investigation Research Object (IRO), is to collect together all the items which relate to the digital object to enable the digital object to be understood. So for data coming from the ISIS facility, the IRO is an aggregation object which collects both the digital object under consideration and artefacts such as raw experimental data, metadata, software, derived data etc. related specifically to an investigation to enable the context of the investigation to be preserved along with the original data from the investigation. Preserving a digital object is complex, but preserving an aggregation which grows over time is more complex. The work described below discusses how an IRO may be built and then preserved. The tools built within the preservation components area are being testing in the Testbed with real metadata from ISIS to establish whether the concept can work effectively with the volume of experiments coming from ISIS.

One of the challenges is to enable this work to move from a very labour intensive process which needs the data producer to be involved to add the additional contextual value towards a more automated process. This is particularly difficult to implement for retrospective processes.

2.3.1 Approach to implementation

For our purposes an investigation is considered to be a unit of work carried out by scientists at a facility. An investigation will have metadata associated with it. It will also have data sets which contain data files, where the data files contain the experimental data. The data files in an investigation can total 10s of GB of data. There can also be metadata associated with the data sets and data files. The prototype uses data from the ISIS facility at STFC. The metadata is stored in a metadata catalogue called ICAT²⁹. Prior to preserving the IRO it was necessary to build them and

²⁷ http://en.wikipedia.org/wiki/Hittite_language

²⁸ <http://www.researchobject.org/>

²⁹ <http://code.google.com/p/icatproject/>

provide a means of manipulating them. The figure below shows the systems and services to enable the creation and preservation of the IRO.

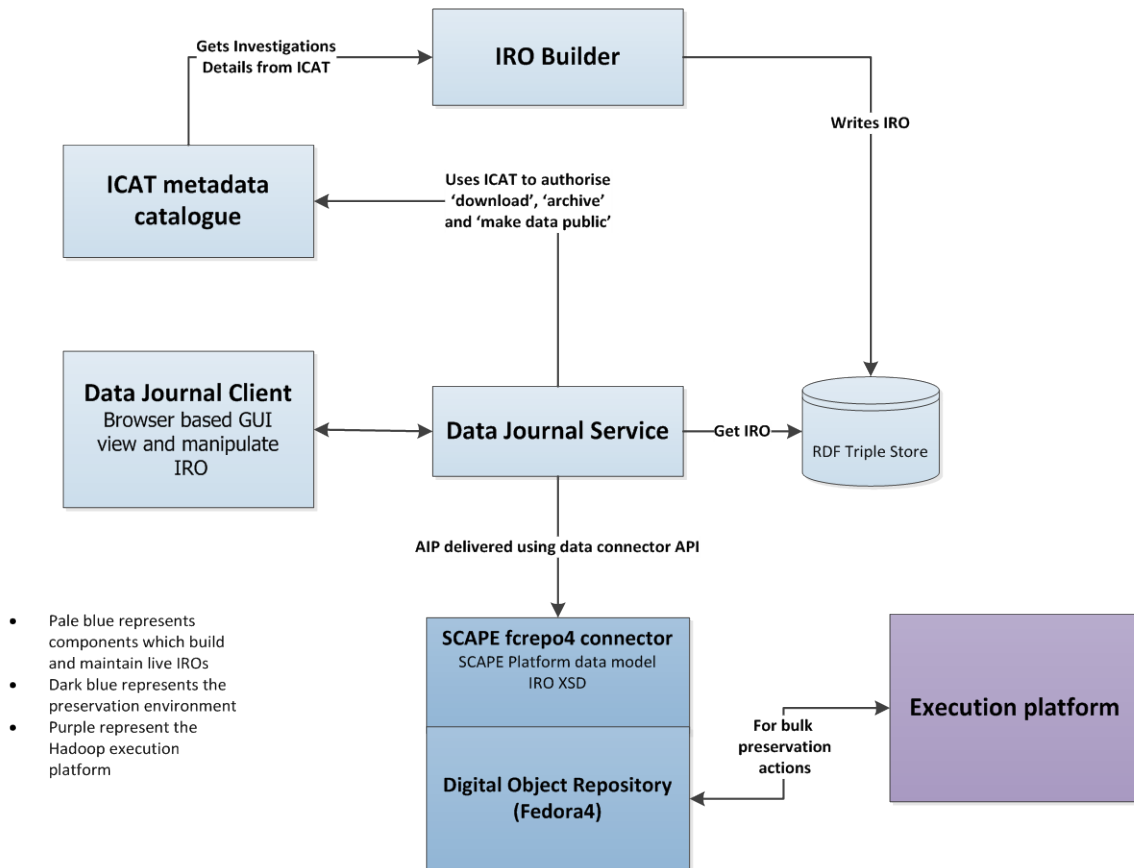


Figure 1: Proposed Architecture Diagram

[1] The IRO Builder

The concept of linked data is at the heart of research objects; therefore it is important that the data be made available in the form of triples. The first step was to create the IRO Builder. This extracts the metadata from ICAT and produces RDF, which is then used as input to a triple store. The RDF makes use of the Core Scientific Metadata Model³⁰ (CSMD). The IRO Builder takes two timestamps as parameters; it uses these in order to retrieve data that was modified between these two timestamps. This makes it easy to keep the triple store up to date with the selected values in ICAT. These metadata about investigations then form the core of the IROs.

[2] The Triple Store

A Fuseki³¹ triple store is used to store the triples generated by the IRO Builder. It is also used to store the annotations (see Adding Context).

³⁰ <http://code.google.com/p/icatproject/wiki/CSMD>

³¹ http://jena.apache.org/documentation/serving_data/

[3] The Data Journal Service and GUI

The Data Journal provides the means to view, manipulate, and preserve IROs. The Data Journal is a GWT³² application, which provides a GUI for users to interact with IROs, and a server to provide data for the GUI and interact with other services.

The current Data Journal viewing interface is designed to enable the inherent structure of the data collection cycles to be navigated. Selecting an investigation brings up a detailed page about the investigation. When developing this page, the current Digital Object Identifier³³ (DOI) landing pages for investigations were used as a starting point. From here it is possible to add context to the IRO and to initiate a preservation action for the IRO.

The basic metadata from the original ISIS metadata catalogue can currently be enhanced with user input of two types of controlled vocabulary: the instrument technique and the analysis software. As the ISIS instrument which has been used to generate the data is known then the PaNKOS ontology³⁴ from the PaNData project can be used to look up the technique or techniques that must have been used to generate the data. This is adding new contextual information as the different instruments at ISIS use different scientific techniques even though they are all using neutrons to probe the structure of matter. The user can then use the GUI to associate one or more techniques with an investigation. This is done by making use of Open Annotations³⁵. An annotation can be used to make an association between objects. An annotation has a body and a target, where the body is defined as the URI of a term from the ontology and the target as the URI of the investigation. In a similar fashion the ISIS provided analysis software which is used by scientists to analyse their data is versioned and each version has its own DOI. Again the user can use the GUI to associate software with the investigation, this results in the generation of a new annotation.

An IRO may evolve over time and depending on the current state of an IRO, different things will need to be validated. The IRO Validator service within the Data Journal Service takes in an IRO plus a validation level. Currently only level 1 is implemented.

- Level 1: check the release date is before now and it has been assigned a DOI.
- Level 2: check that all of the links are valid.
- Level 3: check that there are links to one or more publications.

The checks are meant to reflect the lifetime of an IRO. Calling the validator with a level of 2 will first check that the IRO passes the level 1 validation. At the moment the validator is part of the data journal package. A future release will offer a RESTful interface allowing external tools to validate IROs.

[4] Digital Object Repository: Preserving IRO

It is possible for a user to initiate a preservation action via the GUI. The server will first check to see if the IRO meets the prerequisite criteria via the IRO Validator. Currently this is that the release date of the IRO has passed, i.e. the data as well as the metadata is publicly accessible.

³² <http://www.gwtproject.org/>

³³ <http://www.doi.org/>

³⁴ <http://pandata.org/ontology/>

³⁵ <http://www.w3.org/community/openannotation/>

The IRO, which includes any context linked via annotations, is put in a Metadata Encoding and Transmission Standard (METS)³⁶ package using the SCAPE Data Model. If an annotation from the PanKOS ontology has been made, then the complete object from the ontology is packaged along with data about the ontology itself. This is making the assumption that the ontology to be preserved is not trusted and we also have the rights to preserve it. This is not done for any links to the analysis software as it has a DOI, this means that the publisher has agreed to preserve it, or a reference to it, for the long-term and hence it is trusted. The SCAPE Connector is then used to pass the METS package into a FEDORA repository.

The list of archived versions for an IRO can be viewed via the GUI.

[5] Execution Platform

This component refers to the SCAPE Platform; currently it is not used, but it is there to demonstrate that it could be used for bulk preservation actions on the IROs stored in the Fedora 4 repository, using the APIs produced elsewhere in the project.

2.3.2 Results

The GUI has successfully been used to browse IRO, add context information, download the actual data files, initiate preservation actions and view the list of archived versions of an IRO.

3 BL Contribution

The British Library has contributed two User Stories to this work package. The first was “Normalise Disparate Tabular Data Sources”³⁷ and the second was “Identification, validation and checksumming of a complex corpus”³⁸.

3.1 Normalise Disparate Tabular Data Sources

This User Story aimed to take tabular data that are contained in a variety of file formats (CSV, PDF, XLS, DOC, ...) with none, or non-standard heading names and migrate it into a normalised file format (CSV) for access and long-term preservation. The data set used for this User Story is the UK electoral register. The data is deposited with the British Library by every local authority within the UK but the file format used varies and is not standardised.

A working tool was developed in Java, however, the data set was insufficiently large, not expected to grow significantly and no other data sets could be identified to use with the software. The tool was written to enable easy reuse with other data sets - using a different set of data requires only a new normalisation properties file. The tool comes with built-in MapReduce functionality, and is self-contained for use on a Hadoop cluster. This work was halted due to a variety of factors, with the main reasons being: the data set could not be shared and no open/large data set was forthcoming. The software is publicly available³⁹ and a blog post⁴⁰ was written describing it.

³⁶ <http://www.loc.gov/standards/mets/METSOverview.v2.html>

³⁷ <http://wiki.opf-labs.org/display/SP/Normalise+Disparate+Tabular+Data+Sources>

³⁸ <http://wiki.opf-labs.org/display/SP/Identification%2C+validation+and+checksumming+of+a+complex+corpus>

³⁹ <https://github.com/openplanets/tabular-data-normaliser>

⁴⁰ <http://www.openplanetfoundation.org/blogs/2013-03-01-tabular-data-normalisation-tool>

3.2 Identification, validation and checksumming of a complex corpus

This User Story was started with the aim of identifying, checksumming and validating files in a complex data set. The data set in question is a 1.4TB corpus of UK Ordnance Survey⁴¹ geospatial files collected over several years. This data set is deposited with the British Library for access and long-term preservation and cannot be shared.

The data set contains a vast array of file types. The preponderance of data/files within the data set are limited to two formats: Geography Markup Language (GML) and National Transfer Format (NTF).

The Experiment related to this User Story is still in-development at this time and the following description of it reflects its current state.

Firstly, to better understand the files and the variety of formats within data set, a tool was developed in Java to efficiently generate checksums for the data set (CRC values/MD5/SHA-1/SHA-256) as well as collecting additional information about the files, such as: file size, MIME type from Apache Tika, last modified date and normalised file extension. The aim is that this tool will be modified to generate the same data using Hadoop, such that comparisons of runtime can be made. Work is progressing on adding format signatures to a local copy of Apache Tika, to enable it to recognise geospatial files. The aim is to send these format signatures to the Apache Tika project, for them to consider adding to the upstream codebase.

Secondly, a separate Java tool was created that checks the format validity of the most popular two file formats within the data set; Geography Markup Language (GML) and National Transfer Format (NTF). This software is currently called GeoLint, but may be published with a different name, as part of a larger piece of software.

GML is an XML format, developed by the Open Geospatial Consortium; ISO 19136:2007. This format is used for the majority of the newer data within the data set. The GML files are gzip compressed, therefore to enable correct identification of the MIME type with Apache Tika; the files were first decompressed, if necessary.

NTF is a text based format, administered by the British Standards Institution (BSI); BS7567. This older format is still used for some current data, and also for data held before GML was used.

GeoLint validates the GML files by using an XML validator with the Ordnance Survey Schema⁴². The NTF files are validated firstly by using an external geospatial software library, GDAL/OGR⁴³, to read through the file to verify that the contents are readable. GeoLint also contains internal code that verifies that the header and footer, and line endings of the NTF files are as expected.

The improved format identification and the format validation software will be run at scale when the data set is available on a test Hadoop cluster. To maximise reuse, the code that will be used to run this at scale will be derived from work completed within WP16 (LSDRT).

⁴¹ <http://www.ordnancesurvey.co.uk/>

⁴² <http://www.ordnancesurvey.co.uk/xml/schema/>

⁴³ <http://www.gdal.org/>

4 TUW Contribution

4.1 Persistent Data Citation of Dynamically Created Subsets

Researchers need to be able to reproduce which data sets have been used in an experiment and they need to understand how the decision of using a specific data set has been made. This supports other researchers in verifying the investigations and constitutes good research practice as it enhances transparency. Data citation not only needs to allow the identification of the creators of specific data sets in order to attribute the credit but more importantly tools are needed that help consistent access to the data in a repeatable manner to enable the retrieval of specific subsets and combinations of data sets even in a dynamic context.

Assigning identifiers to data sets allows tracing which data set was used in a study, thus such references to data sets will allow peer researchers to reproduce experiments by utilising the very same data sets again. Most existing data citation concepts originate from the bibliographical domain. They are mainly used to provide persistent identifiers for individual digital objects in a static context. In order to overcome the limitations of entire data set citation, some approaches propose to assign Persistent Identifiers (PIDs) to individual data items, or even individual attributes. While this supports very fine-grained citations, the solution does not scale well as the management of the identifiers becomes too complex.

One investigation may refer to several data sets. Storing exact copies of data set collections does not scale as the storage consumption of large data sets is a limiting factor. Hence a more lightweight method is required which allows researchers to retrieve the specific version of the data sets from the database without the need for data dumps or redundant copies. Furthermore, the data citation mechanism must be transparent for users and support automatic data citation facilities which can be integrated into the scientific workflow.

Also in many cases the underlying data source is still in use and its content is subject to frequent changes, i.e. new data files can be added. This dynamic nature of the research data needs to be reflected in the collections of research data sets which are used by the investigations. The data is accessed by using pointers to the data files; an investigation can use lists of files for including several data sets. As new files can be added, the metadata needs to be stored in a versioned way in the database in order to access a specific state of the metadata at a later point in time. Hence researchers need to reference a specific version of a data set collection which corresponds to the state of the data source which they used in their investigations. By citing this collection and thereby referencing a specific version of the data, researchers can share and retrieve the required data collection again.

4.1.1 Approach to implementation

STFC has implemented a metadata management system which is used for accessing experimental data from every aspect of the research work flow. The system provides an interface where researchers can browse the metadata of the raw data files. In general researchers use a Web frontend called iCAT which supports them in finding the appropriate data set which they need for their experiments. The raw data itself is stored in a different storage system which manages the data integrity and provides access mechanisms in order to retrieve a data set on demand. The search infrastructure operates only on the metadata; hence the subset mechanism which will be utilised in this project is based on metadata. The raw data sets already have unique identifiers which can be



used in order to retrieve them again, but metadata often is not considered as being the basis of persistent identification so far.

Within the scope of this project, we want to extend this system with data citation capabilities which enable researchers to cite and reference data sets and combinations thereof in a more flexible way. We want to trace the way that researchers created their data sets and include versioned metadata for referencing specific versions of the data they used in their experiments. The search functionality that was used by researchers in order to select data sets according to their needs is utilised as a new approach to identifying data sets persistently. In many settings researchers need to make a selection of different data sets which are used together in an investigation. Therefore scientists also need to persistently store the combinations of data sets which they used in one experiment. By storing the queries for the metadata of the data sets, peers can understand how a specific data set was selected. They are then able to evaluate the criteria which have been used by a researcher and therefore deemed relevant for experimentation. This includes not only the identifiers of involved data sets, but also the metadata which was used in order to select specific data sets, thus adding more details about how a selection was made.

The implementation is based upon a query store which maintains records of all queries that have been issued by researchers in order to make their selection of data sets they need for their experiments. The query store saves all query parameters which have been entered by the scientist in order to limit the selection of data sets they are interested in. The metadata which is used in order to describe the data sets is versioned and time stamped. Metadata is never deleted, but flags exist that indicate the status of the metadata and if it is still valid. The list of queries which have been used for such a selection is stored in a time stamped fashion and thus is versioned itself. Hence it is possible to exactly derive which metadata has been relevant for a search query during the time of query execution. The query is matched against the corresponding version of the metadata based on the timestamp of the query execution. This allows for the reproduction of the exact version of records which are needed in order to retrieve the appropriate data sets. Researchers can combine these queries and therefore create a sequence of queries which selects all those data sets they need bundled for the usage in one experiment.

Such a sequence of queries then serves as the reference for selecting all appropriate data sets, thus we assign a persistent identifier for each of these query collections. This persistent identifier can then be used by peers in order to retrieve the selection of data sets again in addition to having the metadata available which was used during the compilation of the experiment.

5 General Conclusions

Careful thought is required before a parallel platform is chosen to be used to speed up processing of big data. The major problems that have been encountered are focused around the filing system of the parallel platform. Considerations need to be made for 3rd party applications. If the data can be streamed to the application such as the postscript to pdf converter ps2pdf, then moving the data from HDFS to the local file system is not required it can be streamed directly to the application and the output streamed back. However, the use of a 3rd party tool such as raw2nexus does not support streaming of data because the file can be huge and streaming is not practical. The whole file needs to be copied to a location where the HDF5 file system tools can be used. While this is effectively a local file system copy (Hadoop and HDFS takes care of running the task where the data is located) this still has significant impact on the node that the task is running on when the file is large.

Can the file that is to be processed be accessed by the parallel system? NeXus/HDF5 has caused problems as it is not easily accessed from HDFS. The file has to be copied to the local file system before access via the NeXus/HDF5 libraries or tools. The movement of large files can cause Hadoop to grind to a halt. Other mechanisms to access the files have been considered on the local Hadoop cluster (with HDInsight there can be no such consideration as these are no other options) such as Fuse and a NFS mounted disk. Fuse proved to be very slow and the systems administrators at STFC would not allow the mounting of NFS shared directories. If an NFS file system was mounted and used it would eliminate the need to copy from HDFS to the local file system (essentially a local to local copy as Hadoop takes the computation to the data) but it would mean that any file operations would take place over a network connection which may not give any performance increase. The same network would be used for all nodes and tasks running on the nodes and this may lead to network contention.

The other major problem is how the data, that is to be processed, is moved from where it is normally stored to where it is to be processed on the parallel platform. The small scale data sets that have been used in the test cases at STFC (~1Tb) took some time to copy onto the parallel platform. The time taken to move the data onto HDFS may outweigh the gains in having the data processed in parallel. Once the data set is processed, the migrated files then need to be moved to some permanent storage. The mechanism for moving data onto and off HDFS is not trivial in STFC. The approach that is taken on the Azure cloud is that the blob storage is available to any of the systems that are hosted on Azure. Movement of data once it has been placed in blob storage is not an issue, it is also replicated and this can be in different geographical locations. A file system like this could be considered highly desirable for most organisations, high availability of data where it is needed (close to any computation) and available over different clusters (different parallel systems HPC, Hadoop...) or virtual machines and geographically replicated.

HDFS is a fault tolerant distributed file system this is to cope with the Hadoop system consisting of a number of components and that each component has a non-trivial probability of failure. This means that some component of HDFS is always non-functional. Applications that run on HDFS need streaming access to their data sets. They are not general purpose applications that typically run on general purpose file systems. It was initially designed to build an index for a search engine. The data is not moved on and off the file system, it may be updated periodically when a web crawler adds updated web pages to the data but it is the final resting place of the data. The user story for migration of raw to NeXus format files, does not fit in with this paradigm well.

There is a vast array of file formats and validation tools for many of the lesser known formats are not readily available. Furthermore the lesser known formats are much less likely to be well understood and documentation may well be sparse or non-existent. Creating tools to validate these formats are likely to take longer and be more complex than for the better known formats. Value decisions on creating these tools have to be made as to whether it is worthwhile to invest the effort, perhaps the effort can be more usefully spent elsewhere.

Developing data citation capabilities for an existing system is based upon several requirements. In order to facilitate citation mechanisms for dynamic data sets, the metadata needs to be stored with timestamps that indicate updates and deletes. The interface then needs to record all criteria which were used during the search process and annotate them with a current timestamp. As a result, the state of the metadata, which was valid during the execution of a search request, can be stored

persistently. This information can then be equipped with a persistent identifier and thus be made available through resolver services. This allows scientists not only to retrieve a specific version of a dataset, but also to understand how the set was constructed and what criteria have been applied during its creation.

6 Glossary

Term	Abbreviation	Description
Apache Hadoop	Hadoop	Framework for processing large data sets on a computer cluster. See http://hadoop.apache.org
HDFS	HDFS	Hadoop Distributed File System. This is Hadoop's file system which is designed to store files across machines in a large cluster.
MapReduce	MR	A programming paradigm for processing large data sets using a parallel, distributed algorithm on a Hadoop cluster.
Investigation Research Object	IRO	A Research Object centred around an investigation
Tool-to-MapReduce Wrapper	ToMaR	A SCAPE developed tool which wraps command line tasks for parallel execution as Hadoop MapReduce jobs
Taverna Server	TAVSERV	Taverna Server is a multi-user service that can execute <i>Taverna workflows</i> . Clients do not need to understand those workflows in order to execute them.
Taverna Workflow		A Taverna workflow is a parallel data-processing program that can be executed by <i>Taverna Workbench</i> or <i>Taverna Server</i> . It is stored as an XML file, and has a graphical rendering.
Toolspec		An XML file written to a standard API that contains details of how to execute a tool for a particular purpose; for example txt2pdf might define how to use a command line tool to convert text to pdf. Toolspecs can have different types such as migration or QA.