Master Thesis

at the Departement of Mathematics and Computer Science

Enabling a Data Management System to Support the "Good Laboratory Practice"

Miriam Ney

Miriam.Ney@fu-berlin.de

Matrikelnummer: 4402119

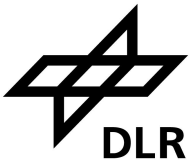
Advisor: Prof. Dr. Lutz Prechelt

2nd Advisor: Prof. Dr. rer. nat. Adrian Paschke

in cooperation with

German Aerospace Center

Simulation and Software Technology



Berlin, 2nd May 2011

Abstract

Conducting experiments and documenting results is daily business of scientists. Documentation enables other scientists to confirm results, reassure interpretations and therefore increase the experiment's credibility. These every day action are regulated and shortly described as: "good laboratory practice".
Due to computerized research systems experimental data get more elaborated, this increases the need for electronic notebooks with data storage and computational features.
The aim of this thesis is to develop a new approach to substitute paper based notebooks. The new approach shall simplify the scientist's work. With the constraint, that it has to stay as evidential and credible as before.
Some of the analysed requirements for laboratory notebooks are traceability of a data item, credibility of an object and preservation mechanisms.
The approach of this thesis is to enable an open source data management system with necessary features for a laboratory notebook. As technologies provenance, digital signatures and secure web services are integrated into the data management system.
This enriched data management system supports the scientist in his daily workwhich helps him to concentrate on research.

Affirmation Statement

I hereby formally declare that the work submitted is entirely my own and does not involve any additional human assistance. I also confirm that it has not been submitted for credit before, neither as a whole nor in part and neither by myself nor by any other person. All quotations and paraphrases but also information and ideas that have been taken from sources used are cited appropriately with the corresponding bibliographical references provided. The same is true of all drawings, sketches, pictures and the like that appear in the text, as well as of all Internet resources used.

Berlin, 2nd May 2011 _____

Contents

Abbreviations

API	Application Programming Interface
BeLab	Beweissicheres elektronisches Laborbuch
DFG	Deutsche Forschungsgemeinschaft
DLR	Deutsches Zentrum für Luft- und Raumfahrt e.V.
ELN	Electronic Laboratory Notebook
GLP	Good Laboratory Practice
ISO	International Organization for StandardiZation
KIT	Karlsruher Institute of Technology
OAIS	Open Archival Information System
OPM	Open provenance Model
PrIMe	provenance Incoroporating Methodology
PTB	Physikalisch Technische Bundesanstalt
RCE	Remote Component Environment
REST	Representational State Transfer
XML	Extensible Markup Language

List of Figures

List of Tables

1 Introduction

With the "Principles of Good Laboratory Practice and Compliance Monitoring" from the OECD research institutes are provided with guidelines to ensure good and reliable research. In it the "Good Laboratory Practice" is defined as "a quality system concerned with the organizational process and the conditions under which non-clinical health and environmental safety studies are planned, performed, monitored, recorded, archived and reported." ([OEC97] p.14) This definition can be extended to other research fields. In the research community being able to prove the quality of ones work is highly relevant for credibility and reliability. Next to organizational processes and environmental guidelines, part of the good laboratory practice is to write and maintain a laboratory notebook. It is usually part of conducting an experiment.
The earliest finding of such a document might be the "Edwin Smith Papyrus" (cf.[Wika], which is a papyrus from ancient Egypt. The papyrus describes rationally and scientifically medical procedures at that time.
Another important person, when discussing scientific methods of documenting experiments, is Galileo Galilei. In his "Dialogue concerning the two chief world systems"[Gal32] he discusses in a dialog of three persons, different experiments, their assembly, the results and the interpretation of this.
The scientific method evolved over time and at last resulted in the before mentioned guidelines to good laboratory practice.
The focus of this master thesis is to describe the prerequisites for a laboratory notebook and the integration of notebook supporting features into a data management system. In the end the relevance of the results to certain user groups should be discussed.

1.1 Working Environment

The thesis is written at the Free University of Berlin (Freie Universität Berlin) in the department for Software Engineering and the German Aerospace Center () at the Institute for "Simulation and Software Technology" in the department "Distributed Systems and Component Software".
"DLR is Germany's national research center for aeronautics and space. Its extensive research and development work in aeronautics, space, transportation and energy is integrated into national and international cooperative ventures"(cf. [DLRa]). The German Aerospace Center focuses its research on the before mentioned four areas: aeronautics, space, transportation and energy. In each area they corporate with different institu-

tions in Germany and Europe. In order to be successful in research, suitable software solutions are needed.

> "Through its cooperation with world-wide leading partners as well as its collaboration in international forums and standardization bodies, the DLR "Simulation and Software Technology" takes an active role in the development of new software technologies and builds up corresponding expertise at DLR for future projects."(cf. [DLRb])

The Institute Simulation and Software Technology develops and identifies new and needed solutions for specific fields, for example in High Performance Computing and virtual reality. It also standardizes support the scientist in issues around software engineering.

1.2 Motivation

As computer aided experiments get more powerful, the generated data gets more elaborated and voluminous. Therefore handling this data is increasingly complicated. In order to get hold of the situation, the German Aerospace Center developed the open source data management application "DataFinder"[Data]. This data management system is supposed to help the researcher to manage their data. It allows heterogeneous storage backend, flexible extensions to its interfaces and meta data support.
The next step is to extend the DataFinder in such a way it can be used as a tool to support the good laboratory practice, or in different phrasing: as an electronic laboratory notebook.
The master thesis should conclude that extending the data management system to an electronic laboratory notebook is possible. It should also show concepts and an implementation of a prototype meeting the main requirements for an electronic laboratory notebook. If the DataFinder is able to fulfill these criteria it could help many researchers to simplify and improve their work.

1.3 Distinction to other research fields

This master thesis distances itself from the following research areas:

Management of scientific workflow A scientific workflow is the procedures a scientist does to process data from raw data to evaluated data or as Ludäscher et al. put it:

> "These are networks of analytical steps that may involve, e.g., database access and querying steps, data analysis and mining steps, and many other steps including computationally intensive jobs on high performance cluster computers. " [LAB+05]

This thesis does not implement strategies to implement such a scientific workflow with access to different evaluation software or other systems. This thesis focuses on a documenting approach of the data. The information is stored and made accessible to others. A system of the DLR that supports the integration of different scientific workflows is for example [rce].The possibilities of integrating provenance with scientific workflows were investigated in [DF08].

Research on provenance This thesis does not focus on the different architectures of provenance systems [GMTM05], nor different approaches to implement provenance. The thesis incorporates an available system and adjusts it in a way it can be used generally. Further information on provenance in science gives Simmhan et al. in [SPG05b] and in [SPG05a]. In the first paper Simmhan discusses different techniques for data provenance. The second paper the use of data provenance in e-science is presented.

Developing strategies for long term preservation Another research field that is being touched is long term preservation. This thesis implements a service providing long term preservation [BeLa]. No further inquiries on the strategies and concepts behind it will be made or stated. Information on long term preservation gives the ISO reference model on OAIS in [SDS02], which is shortly described in chapter 2. The DataFinders capability according to this is stated in chapter 3.3.2.

1.4 Structure of the thesis

The general structure of the thesis starts with background information in the first chapter, where general requirements for the "Good Laboratory Practice" and a laboratory notebook are analysed. Based on the requirements models and processes for the data management are designed. The data management system DataFinder is then adapted to the requirements and models. Subsequently the implementation is evaluated. In the end a conclusion of the whole system and thesis is derived.
The following describes the chapters more detailed:

Chapter: Background In this chapter background information is given. More information on the current provenance research and its practical approaches are described. In detail analyzing a provenance use case () and modeling () approaches are explained. At last several archiving strategies are discussed and the aims of the project are presented. At last scientific data management is explained. Also the data management system DataFinder, which is used to support scientists, is introduced.

Chapter: Implementation This chapter is divided into three sections. Each explains one step of extending a data management system to a laboratory notebook.

Section: Requirements The requirements in this chapter evolve around the laboratory notebooks. International and national regularities, as well as main literature, are analysed and summarized. A next step compares different existing electronic laboratory notebooks and each notebook is evaluated.

Section: Concepts Two models are the main result of the chapter on concepts. The first model is for the data management system and maps the requirements from good laboratory practice into a hierarchical model with describing meta data. The second model deals with the data life cycle in a scientific experiment. It is developed based on tools used for data provenance.

Section: Implementation of concepts This chapter focuses on implementing the concepts. The data management system DataFinder is customized to meet the analysed requirements. Three main characteristics, which are missing, are identified: a feature for evidential durability, for signing data and for evaluating the origin of data. Each feature was implemented with a different iterative approach. The steps of integration are explained in this section.

Chapter: Evaluation At last the different implementations into the data management system DataFinder are evaluated. Each developed script and the whole resulting application is evaluated for their usability, adaptability and adjustments.

Chapter: Conclusion The conclusion wraps up the results of the thesis; as well as its impact on the scientific community. Furthermore it gives an outlook on still needed work on the subject.

2 Basic concepts from related research areas

To understand some concepts described within the thesis, this chapter gives background information on relevant research areas and their application.

2.1 Motivation for good laboratory practice

Due to several discrepancies between denoted and actual results in scientific publications the OECD decided 1997 to update the "Principles on Good Laboratory Practice"(cf. [OEC97]) from 1978. The introduction clarifies these intentions:

> "The principles of Good Laboratory Practice (GLP) have been developed to promote the quality and validity of test data used for determining the safety of chemicals and chemicals products." (cf. [OEC97] p.13)

These principles suggest how to conduct and setup experiments in scientific research environments. They also give recommendations on the environment for scientific work. The principles explain how those studies should be "planned, performed, monitored, recorded, archived and reported." (cf. [OEC97] p. 14)
Although the principles focus on chemical and non clinical health studies, they can easily be adopted to all research areas. This has been done by the DFG, who have published their own recommendations on good scientific practice (cf. [DFG98]). These recommendations is a set of rules for scientific work that " are designed to provide a framework for the deliberations and measures which each institution will have to conduct for itself according to its constitution and its mission." (cf. [DFG98] p.50)
Both papers cover in their regulations several scientific areas. They state rules for the group of scientists, the organization they work for and for the individual. The framework gives guidelines for quality management, data archiving, standard operation procedures and the execution of studies. In addition to this, responsibilities of each participant are defined. Further the principles give recommendations on which information should be known of a test system, the facility, and the test items.(cf. [OEC97] pp. 18).
The laboratory notebook is a part of the good laboratory practice documents everything happening in a study.

2.1.1 Data management according to the good laboratory practice

In order to comply to the the OECD requires that several documents are available to the researchers. For example manuals and standard procedures for all test systems and apparatus are necessary. For a study the OECD requires a detailed study plan, which is

> " a document which defines the objectives and experimental design for the conduct of the study and includes any amendments." (cf. [OEC97] p.15)

The data needed for GLP can be divided into two groups. Data needed to use and operate the instruments, belongs to one group and data unique to one study to another group.

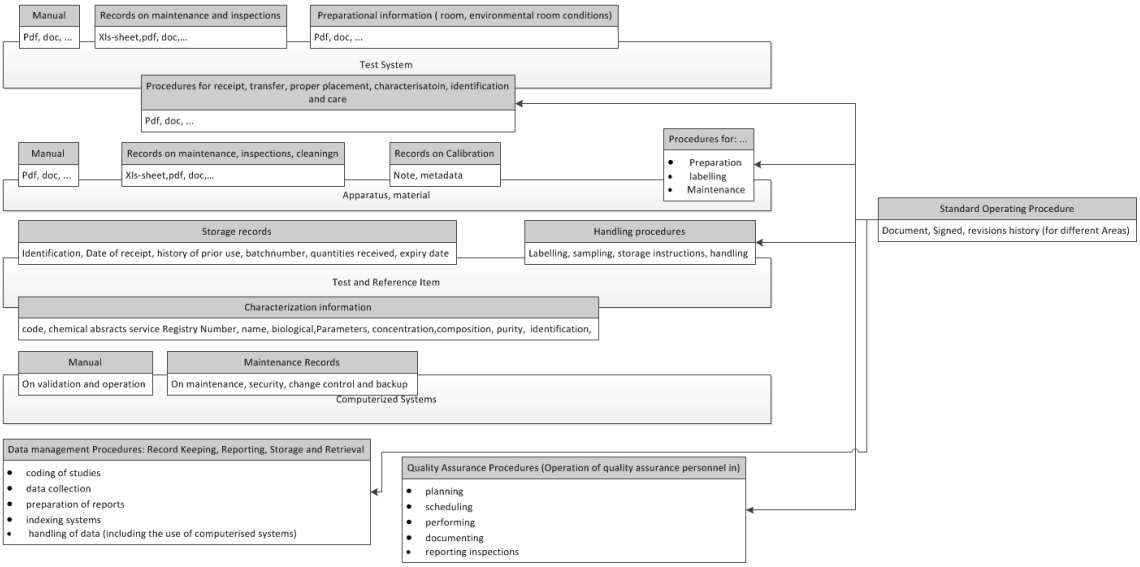


Figure 2.1: General data, which is needed for the good laboratory practice

In figure 2.1 the general data from the first group is listed. This group consists of different subgroups: Test System, Apparatus, Test Item and some others. Each subgroup requires specific documents (cf. [OEC97] p. 22-25). For example in order to control a result in an experiment the correct calibration information needs to be available. Those information also inform the scientist of possible defects w ithin an apparatus and help to interpret certain results. If all the information is generally available it is easier for new scientists to get used to procedures. This enables them to hold up a level of quality. The second group is formed by data unique to a study. It centers around a study plan. The study plan can contain references to data items from the general data group. The group itself consists of data generated during a study. The study plan, ideas, results and evaluations, are usually documented in a laboratory notebook. The GLP requires further the generation of a specific report for each study. This report depends on the data entered in the laboratory notebook.

The last requirement for data management stated in the OECD and the DFG recommendations revolves around "Storage and Retention of Records and Material" (cf. [OEC97] p. 29). Any data somehow related to the study, including manuals and standard procedures, but especially raw and experimental data, need to be archived. The content of the archives need to be preserved for a certain time.

2.1.2 Identifying the workflow of an scientist

From the before mentioned recommendations a workflow including a data life cycle can be extracted. In [OEC97] starting on page 25 a whole section explains how a study should be performed. It focuses on a study plan, which defines the steps and around which the experiments are designed and conducted.
Figure 2.2 presents the workflow and the generated files for each step.

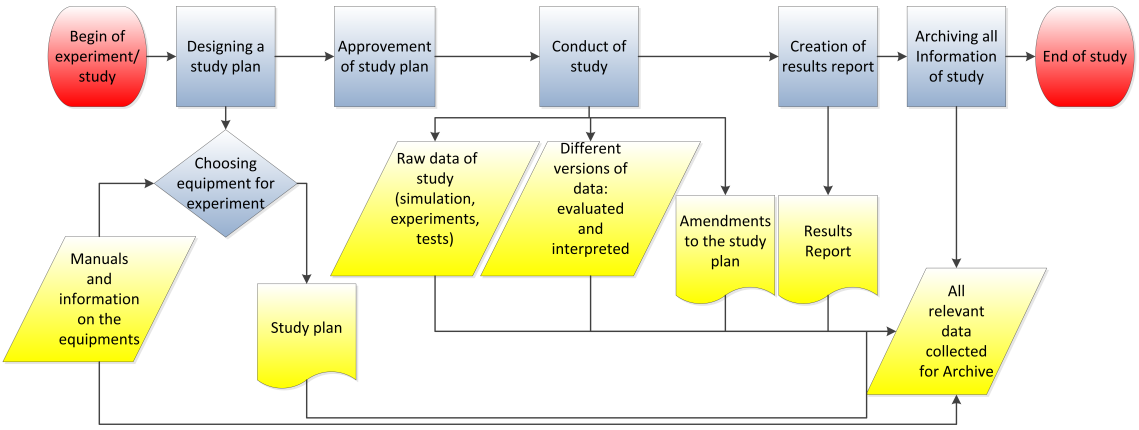


Figure 2.2: Workflow of a scientist according to the "good laboratory practice"

The workflow starts with design of the study plan, where one chooses equipment, collects information and writes the study plan. The study plan needs to be approved by an authority. This whole process can be summed up as preparation phase. The next step is the execution of the study plan. In this step most data is generated. First the raw data that comes directly from the equipment. Then, as the study proceeds, due to evaluation and interpretation different versions of the data are produced. During each phase the study plan can be amended, because news need to be integrated. Next a report on the study is created, this report summarizes the data from phases before and interprets the results. At last all data that is important for the study is archived.
In a first working package of the BeLab project the workflow and the generated data in a scientific study was analysed.(cf. [BeLb]). First the example of the Max-Planck-Institut für Dynamik und SelbstorganisationIt is introduced. Figure 2.3 shows common experimental data, when investigating turbulence in a wind tunnel.
At first physical parameter settings are documented as preparation. During the execution for example raw data, like pictures of accelerated nano particles, are produced. When evaluating the pictures trajectories of nano particles can be generated. The eval-

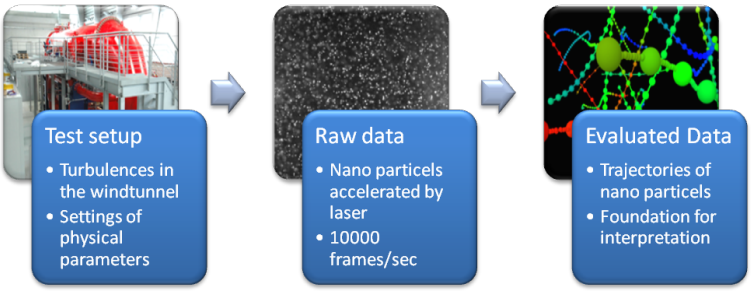


Figure 2.3: Example data workflow (adapted from [BeLb])

uated data is the foundation for the following interpretation.
Each phases is defined abstractly during a workshop. Figure 2.4 shows the presented scientific workflow.

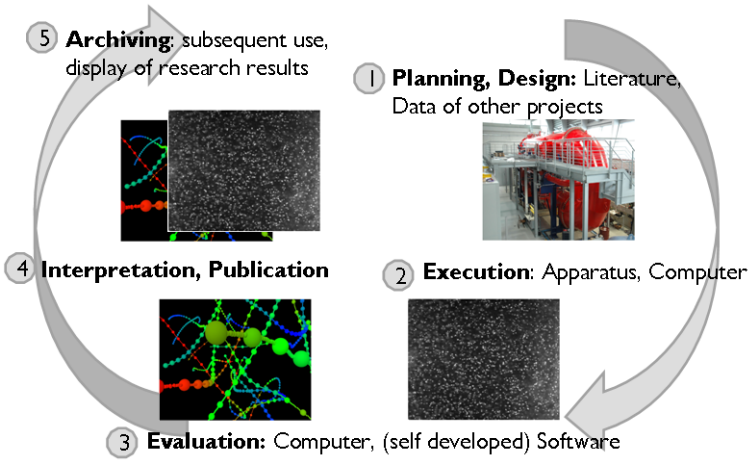


Figure 2.4: Scientific data life cycle (adapted from [Pot11])

Here is similar to the [OEC97] principles a differentiation between preparation, execution, evaluation, interpretation and archiving.
The results from the research in the BeLab project show, that the principles defined by [OEC97] and [DFG98] are lived in the scientific environment. These workflows are the foundation for the processes and models defined in chapter 3.2.

2.2 Provenance

provenance originates from the Latin word: “provenire “ meaning “to come from” [MW10]. It is described as “the place that sth. originally came from” thus the origin or source of something (cf. [Weh00]). It was originally used for art objects, but

other disciplines adapted it for their objects, such as fossils or documents. In the field of computer science and data origin it could be defined as:

> "The provenance of a piece of data is the process that led to that piece of data." [Mor10a]

Based on this understanding approaches for identifying provenance use cases, for modeling processes and for integrating provenance into applications are developed. Also concepts to store and visualize provenance information are investigated. An overview of the different areas of provenance gives figure 2.5.

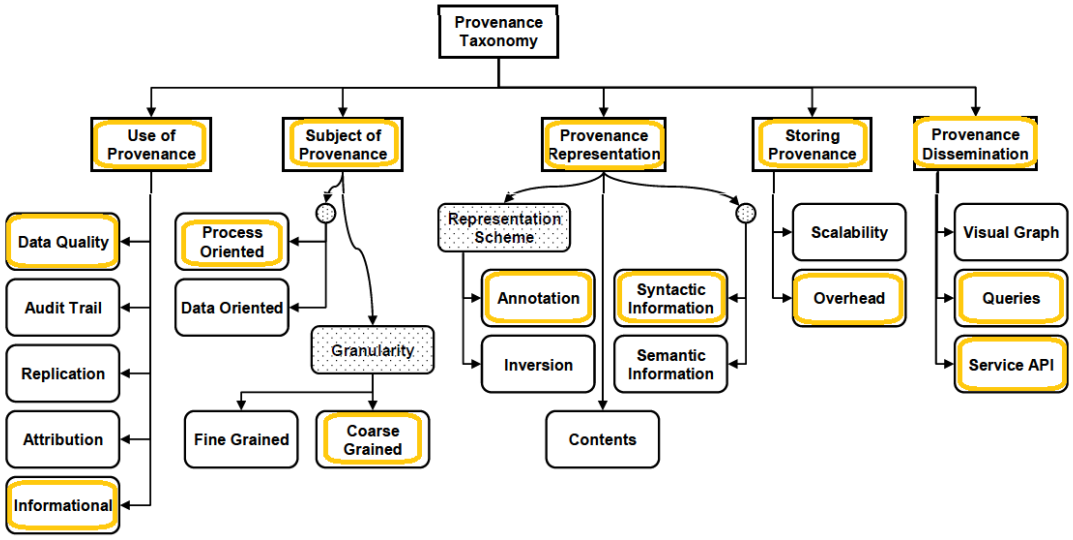


Figure 2.5: Provenance taxonomy according to [SPG05b]

The figure shows five major areas: Usage, Subject, Representation, Storage and Dissemination. [SPG05b] gives a detailed description on each area and their subdivisions. In this master thesis provenance enables the data management system DataFinder to provide information about the chain of data items leading to a data item. The following list fits the thesis use case into the taxonomy from figure 2.5:

Use of provenance provenance is used to present *information* of the origin of the data, but also to provide *data quality*.

Subject of provenance The subject is the *process* of conducting a study or experiment. It is focused on the documentation of it. To identify the subject further the provenance Incorporating Methodology (PrIMe) is used.

provenance Representation The provenance will be represented in an *annotational* model, based on the Open provenance Model (OPM) and it will mainly hold *syntactic information*.

Storing provenance The provenance information will be stored in the "noblivious" system, which can hold more information, than necessary.

provenance Dissemination To extract the provenance information, the provenance system can be queried via a "Gremlin" interface.

The main concepts of PrIMe, OPM and the provenance system noblivious are described in the following sections.

2.2.1 PrIMe - Provenance Incorporating Methodology

Munroe et al.([MMG+06]) developed with PrIMe a methods, which enables applications to identify parameters, that can be used to answer provenance questions. A provenance question usually identifies a scenario, where provenance information is needed.
The overall structure of PrIMe is shown in figure 2.6.

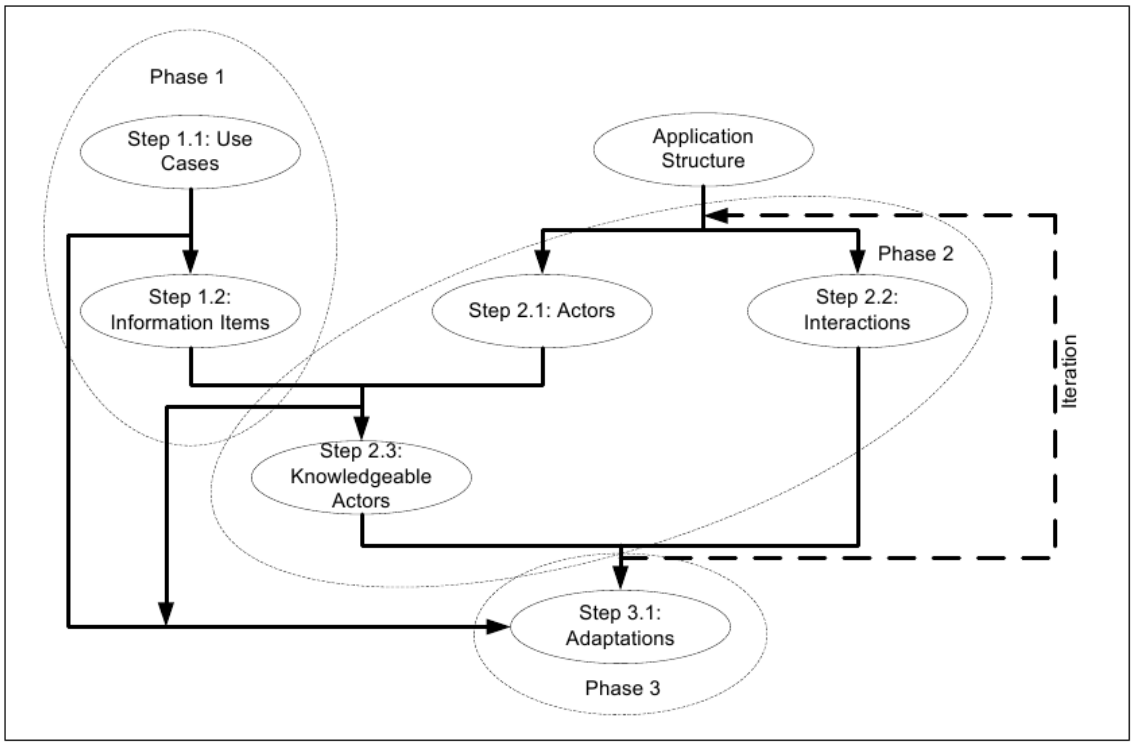


Figure 2.6: Structure of PrIMe approach [MMG+06]

This approach was adapted in ([Wen10] because it used the p-assertion protocol([Wen10] p.15 and [MMG+06] p.2). The p-assertion protocol is similar to the used OPM and can be easily adapted. The following list describes the three phases of the adapted version:

Phase 1 "In Phase 1 of PrIMe, the kinds of provenance related questions to be answered about the application must be identified"([MMG+06] p.7). So first provenance Questions are defined. Then corresponding data items, that could provide the answer, are investigated.

Phase 2 Subprocesses, actors and interactions are identified in phase 2. The subprocesses are part of the adaptation in Step 2.1. The actors generate data items

or control the process. The relations between subprocesses and data items are defined as interactions (Step 2.2). Actors, processes and interactions are modeled with OPM.

Phase 3 The last phase finally adapts a system to the provenance model. In this phase the provenance store is filled with the information from the application.

2.2.2 OPM - Open Provenance Model

The Open provenance Model[OPM] is the result of a provenance challenge to provide a format that can be used as interchangeable format between provenance systems.
In its core specification it defines elements, such as nodes and edges, that can be used to describe a provenance system. Figure 2.7 shows the available relations and elements.

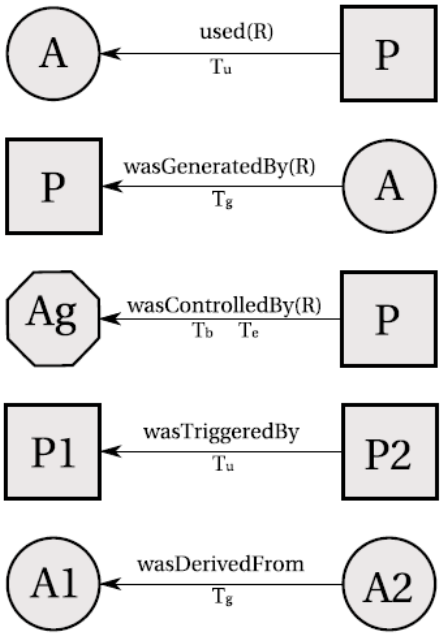


Figure 2.7: Edges and nodes of the OPM from [MCF+09] p. 15

Nodes can be processes (P), agents or actors (Ag) and artifacts or data items (A). The nodes can be connected through the edges: used, wasGeneratedBy, wasControlledBy, wasTriggeredBy and wasDerivedFrom. Each edge has different nodes as source and target, clearly defining the possible relations within a provenance model. Each node can be enriched with an annotation.

In figure 2.8 an example for writing a text shows the usage of the models notation.

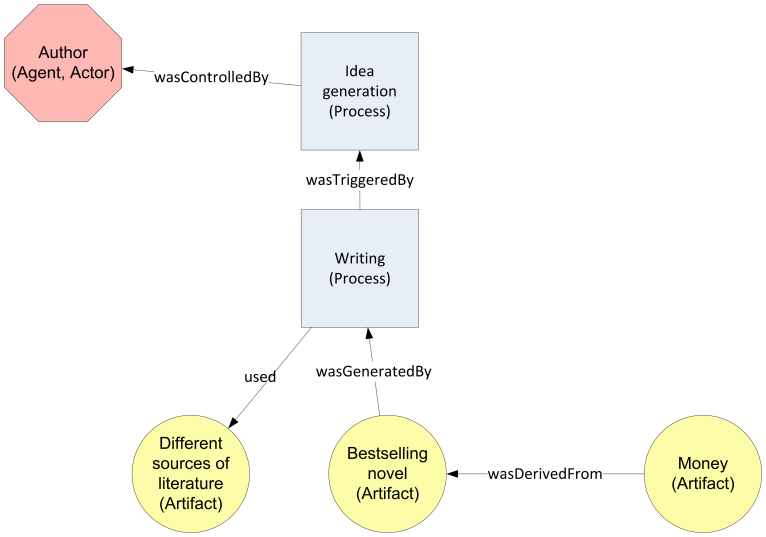


Figure 2.8: Example of writing a text as OPM model

The example is, if an author(actor) has the idea(controls the process of thinking and inspiration) of a good story, so he starts writing it (triggered by a good idea). For it to be a credible novel(generated from writing), he needs(uses) literature to research his idea. If it is a good novel, money can be made(derived) with it.

2.2.3 "Noblivious" - a provenance system

Groth et al. describe in [GMTM05] theoretically the architecture of a provenance system. The figure 2.9 shows the main idea of it, which is also the idea behind noblivious.

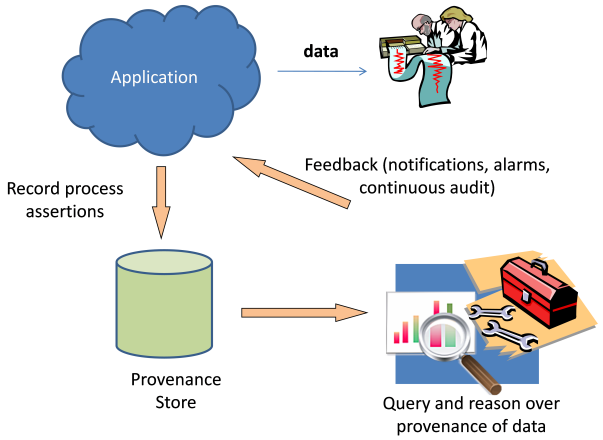


Figure 2.9: Representation of a provenance system from [Mor10b]

In this scenario a provenance aware application sends information of interest to the

provenance store. From this store inquiries and information is gathered, and possibly given back to the application.
To record the information, different approaches have been investigated. In [HBM+10] four different realizations are discussed: Relational, XML with XPath, RDF with SPARQL and semi structured approaches. They conclude that semi structured approaches are the most promising. When using a semi structured approach, the used technology has no underlying formal structure but has some way of being queried.
For this thesis the storing system of "noblivious" is used. This system, which was developed for modeling software engineering processes in [Wen10], uses a semi structured approach. It uses the graph database "neo4j"[neo] and as querying language "Gremlin"[gre]. Further it provides a REST interface to load data into the store, and a web front end to query the database.
It is not the first implementation that uses a graph database as storing technology, in [TC09] it has been proved sufficient.
This system was used, because it was a prerequisite.

The graph database: neo4j

> "Neo4j is a graph database, a fully transactional database that stores data structured as graphs."(cf. [neo])

Graph databases like Neo4j have the advantage, that they are flexible. The flexibility makes it possible to develop databases fast. Neo4j is dual licensed as AGPLv3 and commercially.
The combination process of using neo4j and to model OPM is described in [Wen10]. Figure 2.10 shows the previous example as a graph. The figure uses a notation which combines neo4j with gremlin[gre].

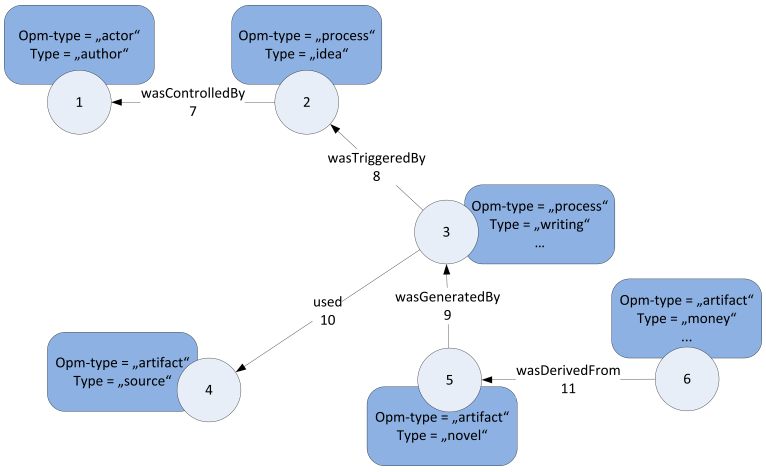


Figure 2.10: OPM example in neo4j

For each OPM element a node (vertex) in the database is produced. The nodes are

indexed according to the neo4j standard. Each node (vertex) can be extended with further information, for example the OPM specific information. The edges that connect the nodes(vertex) are indexed as well and can have added a label, which in this case is the OPM relation.

The query language: gremlin

"Gremlin is a graph traversal language."[gre]. Gremlin provides an interface to interact with the neo4j graph database.
An example for using gremlin with neo4j on the example database:

```
$_g := neo4j:open('database')
$authors := g:key($_g, 'type', 'author')
$authorX := g:key($authors, 'identifier', 'authorX')
$books := $author/inE/inV[@identifier']
```

The query searches for the names (identifiers) of all ideas a certain authorX had.

2.3 Scientific preservation

Results and data of experiments are valuable for publications and future research.This makes preservation of data is an important for scientists.
The gives in his Recommendation 7 the following advice:

> "Primary data as the basis for publications shall be securely stored for ten years in a durable form in the institution of their origin." ([DFG98] p. 55)

The commentary following the recommendation explains the importance of owning the primary data of a publication. The archives enable other scientists to reproduce findings and therefore proof validity. Especially in social sciences this is already habit. In Germany the nestor group [nes] works on solving issues of long term preservation from the perspective of several areas.
Storing data, that is written on paper, is been exercised frequently and somewhat easy. When trying to archive digital data, the procedures are currently more difficult. In the next two sections current strategies for preservation systems are presented first. Then a German project, which focuses on long term preservation as service including an evidential component is described.

2.3.1 OAIS - Open Archival Information System

> "An OAIS is an archive, consisting of an organization of people and systems, that has accepted the responsibility to preserve information and make it available for a designated community" [SDS02]

As the citation above states an is an archive for long term preservation of information. On the one hand the OAIS is an archive, but on the other hand OAIS stands for the Standard 14721. The OAIS provides a reference model for an OAIS.
The reference model provides a framework for understanding and applying long term preservation and archival concepts. It enables communities to preserve their most valuable digital data.
It states that an OAIS must be a system, that ensures that the data within is readable, even if the system ceases to exist. So in order to become an OAIS several documents and standards need to be met.
In this master thesis the OAIS is touched, when discussing the preservation of data in a laboratory notebook.

2.3.2 BeLab Project - Evidential long term preservation

The BeLab - Project is a German project focusing on evidential long term preservation (cf.[BeLa]). It develops a concept for storing data long term and ensuring it to be evidential valid[1].
The project is funded by the German Research Association (DFG) and consists of three main member organizations: , University of Kassel and the . Each member has expertise in at least one of the following areas: law, cryptography, scientific research and software engineering. The project started beginning of 2010 and continues until 2012. During this time period they develop a concept for evidential documentation of primary data of research experiments. The concept also involves long term preservation. Currently the project implements the concept into a prototype.
After evaluating different implementations of electronic laboratory notebooks and scientific data management systems the project considers the DataFinder suited for testing their service. The necessary extension is described within this thesis.

2.4 Scientific data management

When analysing the data management situation at scientific research labs, several problems are noticed. First each scientist is solely responsible for the data he generates and can manage it as he wants to. That way others can not access it, and duplicate work occurs. A second problem is: if a scientist leaves the organisation no one understands the structure of the data storage. The results are lost. Third a lot of researchers spend a lot of time searching for data. This makes them lose a lot of time. making them unproductive. Last: Due to long archiving periods and more generated data the data volume increases.
To handle this situation, that is common in different research institutes, the DLR fa-

[1]Since the project is a German project evidential means in accordance to German law.

cility Simulation and Software Technology developed the scientific data management system DataFinder(cf. [Data].

2.4.1 Data management with the DataFinder

The DataFinder is an open source software written in Python. It is uses a server and a client component. The server component holds data and meta data and with the client the data is accessed and managed. The managed data is also called the shared data repository. The figure 2.11 shows the user interface of the DataFinder, when one has not been connected to a shared file repository.[2]

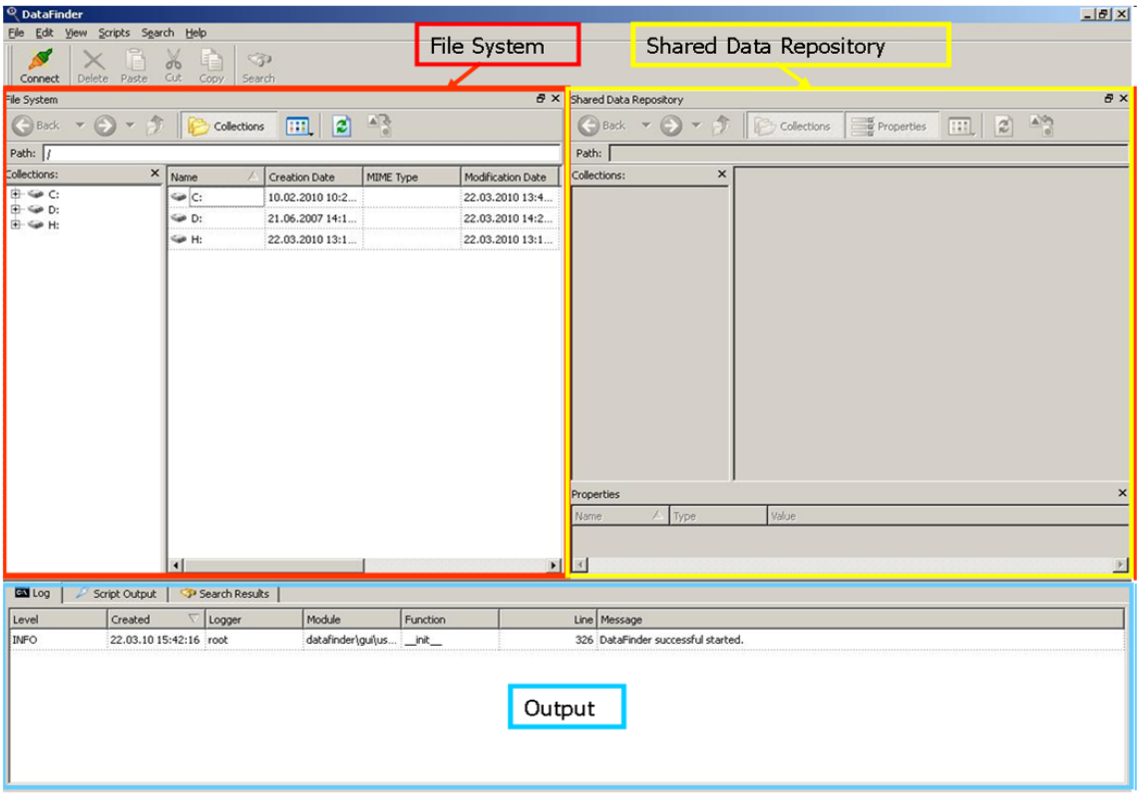


Figure 2.11: User interface of the DataFinder

It is designed similar to a file system. On the right side the local files are displayed and on the left side shared files are displayed. Common actions are possible on both sides: opening, copying, pasting, importing and exporting data. Shared data can be extended with meta data.

Advantage of the system is, that an individual data model is integrated into the application and must be followed by any user. The data model defines a structure of collections. The collections can have different data types, that give the user an idea of what to insert into that collection. The data model also defines meta data items for

[2]Picture 3.9 shows a connected shared repository.

collections. The meta data can be defined either as optional or mandatory information when importing a data item. Based on the data model the data is managed on a heterogeneous storage system.
The second advantage is the ability to manage data on different storage systems, while having one interface. The user can load data on different storage systems, for his view on the data this is not important. Data, which is entered on one system can be managed in the same collection as any other data item.
A third advantage is the possibility to extend the whole application with python scripts. It enables the user to use more customized features, such as tool integration.
The data management system targets to leave many options open and so to be highly extendable for many fields. The DataFinder is already used in different research fields. New use cases are identified and implemented constantly. One of this new use case is the support of the good laboratory notebook from this thesis. In chapter 3.2.2 the data model for the use case is described and in chapter 3.3 the script API is used for implementing new features.

3 Extending a data management system: Requirements, Models and Implementation Concepts

This chapter describes the possibilities to extend a data management system to support the good laboratory practice. The first section deals with a requirements analysis of a laboratory notebook. Next models needed for the configuration are developed and explained. The last section describes how a choice of requirements can be integrated into the data management system "DataFinder".

3.1 Criteria defining an (electronic) laboratory notebook

> "Das Laborbuch ist ein Tagebuch des experimentierenden Naturwissenschaftlers" (cf. [EBG06] chapter 1.3, page 16) [1]

This section details the above mentioned laboratory notebook, and the common usage. On the one hand the laboratory notebook is used for the documentation and on the other hand it is the source for writing reports and publications. Often it is not explicitly conducted as a laboratory notebook, but equals a collection of files and notes. It can be generalized as a scientific documentation.
The structure of this section is first a requirements analysis based on different literature. Then electronic laboratory notebooks are evaluated. At last examples of scientific documentation are shortly described.

3.1.1 General requirements for a laboratory notebook

To have a correctly maintained laboratory notebook means being able to prove ideas and results. In order for laboratory notebooks to have evidential value it needs to fulfill more requirements on authenticity. General requirements are listed in the following, the order is not relevant.

Chain of events If reconstruction of an event is needed, to know the sequence of actions and entries can be of great value. This requires of a laboratory notebook to

[1] The laboratory notebook is the diary of the experimenting scientist

display some sort of history, for example with numbered pages and dates on each entry.(cf.[EBG06] p.18)[2]

Preservation To be able to support the results described in a publication, any data leading to the result needs to be archived for a specified length. (cf. [EBG06] p.18 [3] and [Pas10])

Genuineness Another part of being authentic is being genuine, such that there is no possibility to temper with data as soon as it is entered into the laboratory notebook. (cf. [EBG06] p. 18 and as "Signierbarkeit" in [Pas10])

Immediate documentation For authenticity immediate documentation of all records is necessary. Any notes should be entered into the laboratory notebook, not on a whiteboard or some other temporary medium. This requires of an electronic laboratory notebook to be available at any moment. (cf. [EBG06] p.18) [4]

Protocolling This requirement specifies the style in which the laboratory notebook is held. It means entering short descriptions with all necessary information such as settings of an apparatus, should be possible.(cf. [EBG06] p. 22)

Short notes For a higher value of a laboratory notebook it should be possible to comment easily on a result or make a quick note of an idea that came to mind. A report can profit from these notes, which might be the core finding. (cf. [EBG06] p.16)

Verifying results For more credibility, other scientists need to attest research results. Also a scientist needs to be able to verify his own results. This means a mechanisms needs to be found to "witness" to a result.(cf. [EBG06] p. 19) [5])

When maintaining a laboratory notebook electronically, it is possible to have more support. The features support data organization and extend the possible usage scenarios. This are features, such as collaboration and searching inside of data. Those features are listed in the following.

[2] "Die Seiten des Laborbuchs sind zu nummerieren. Jedes neue Experiment beginnt auf einer neuen - nicht notwendig der jeweils nächsten Seite und trägt ein Datum.": the pages of the laboratory notebook are to be numbered. Each new experiment begins on a new - not necessarily next page and has a date.

[3] "Laborbücher sind gebundene Notizbücher mit festem Einband und gutem Papier. Geschrieben wird mit Kugelschreiber, dadurch werden Einträge dokumentenecht": Laboratory notebooks are hard back note books with good paper. Written with ball pen, it makes the entries indelible.

[4] "Die Aufzeichnungen müssen sollen sie authentisch sein, sofort zum Zeitpunkt der Beobachtung oder Durchführung eingetragen werden": The records in order to be authentic need to be recorded the moment they are observed or executed.

[5] "Wichtige Versuchsergebnisse werden von einem Kollegen durch Unterschrift bezeugt": Important research results are being testified by a colleague

Accessibility of software ressources The minimal requirement is the general availability of a documenting system to a laboratory. Furthermore the price or possibility to have access to the source code depends on the situation.

Collaboration For better and more valuable results it often helps, when several people can discuss an idea or results. This requires of the notebook system to have a collaboration mechanism. This requirement extends the requirement: "verifying results". (cf. [EBG06], [Pas10])

Device integration The integration of devices has the advantage of capturing data automatically. The integration should directly store the data coming out of a device in the correct location within the documentation system. The integration helps to prevent forged results and supports the requirement: "immediate documentation".(cf. [EBG06])

Enabling environmental specialisation Electronic laboratory notebooks should be fitted to the user and its research field by integrating customized features. For a general laboratory notebook one should only integrate basic features, but leave the possibility to integrate more features.

Flexible Infrastructure To ease into the usage of a system and helping it to spread, the setup should be as simple as possible. The components needed for the system should fit into the infrastructure of a laboratory.

Individual Sorting In some cases different views of the stored data is needed. So sorting by categories or filters over data adjusted for the situation can be helpful.

Rights management Electronic laboratory notebooks need to specify rights management mechanisms, because in contrast to other laboratory notebooks it is not protected by person. It should be regulated who is able to read and write an object. (cf. [Pas10])

Searchability The digital management of data has the advantage that searching data is easy. Therefore searching mechanisms should be integrated into laboratory notebook system. It can increase the scientists productivity, when he can find prior results from himself and colleagues quickly.(cf. [Pas10], [EBG06])

Variety of dataformats The more data formats an electronic laboratory notebook system supports the more universal is its application area. Further scientists can work with electronic notebooks more interactively with their data. (cf. [EBG06], [Pas10])

Versioning Versioning means having different versions of one document, therefore one is able to compare versions. It means a mechanism as it is available in version control systems such as subversion is needed for the electronic laboratory notebook.(cf. [EBG06] p. 26 and [Pas10])

Electronic notebooks not only improve the scientist work, but can have probelms, that need to be avoided :

Authenticity in general Proving the authenticity of information saved on an electronic medium needs to be solved in an electronic laboratory notebook. The identity of the author or the results presented need to have valid confirmation of their originality and authenticity to be credible. Ensuring this is only possible with valid and fitting cryptographic mechanisms.(cf. [EBG06] p. 30 [6])

Complexity A laboratory notebook should be simple to use, easy to understand and data entered effortless. An easy start lowers the barrier to change from conventional ways to electronic notebooks(cf.[Nbm] and [EBG06] p. 30). Fulfilling requirements such as "protocol style" helps to avoid this problem.

Integrity Ensuring integrity when using electronic storage mediums is important, since it "seems" easier to forge results when saved electronically. To prevent forgery the software engineer needs to implement correct methods and be careful with his design. (cf. [EBG06] p. 30)[7]

Elimination of scepsis When introducing new systems it is hard to convince prospective user of the advantages of the new system. Having reservations against the new system, in respect to security and other issues are common, mostly not based on facts. A way around this is the full support of the management level. But even this cannot guarantee the success of a project.

As conclusion for any implementation of an electronic laboratory notebook it can be said:

> "It is important to remember while implementing an electronic notebook to ensure an acceptable level of simplicity. If the electronic notebook is too complex or too different from a paper notebook, then it is unlikely to catch on with the scientific community." [Nbm]

All mentioned requirements and features, characterize a laboratory notebook and are the basis of the following comparison of electronic laboratory notebooks. Further they will be used in the end to evaluate the suitability of the developed system.

3.1.2 Comparison of electronic laboratory notebooks

This section compares different electronic laboratory notebooks according to the requirements specified in the above section. The notebooks are divided in a table for

[6] "Kritisch stellt sich die Frage nach der Authentizität der Mitteilungen im elektronischen Laborbuch": Critical is the question of authenticity in electronical laboratory notebooks.

[7] "Der Umgang mit nachträglichen eingefügten oder veränderten Daten, liegt in der Verantwortung der Wissenschaftler und des Software Entwicklers geeignete Lösungen zu finden.": The handling of additional or changed data, lies in the responsibility of the scientist and the software engineer to find adequate solutions

commercial and non commercial applications.
The comparison is done on free and available information and documentation found about some laboratory notebook implementation and does not claim completeness. The notebooks were chosen for diversity, accessibility, information access and being up-to-date. The tables including the information of the comparison can be found in the appendix A. The comparison includes commercial and non-commercial software. Each of the notebooks from the comparison could be tested intuitively and mostly followed the requirements defined earlier. The evaluated notebooks and the results of the evaluation were:

Open Enventory (OSS) Open Enventory [ope]is a laboratory notebook which is adapted to its chemical environment.It has a test item database connected to it, which is important for the chemical field. All in all it is not very secure, passwords are send in the url. When testing it on the provided server, it was very unstable. For a general laboratory notebook it is too specific and although it is open source the code could not be downloaded.

EMSL ELN (OSS) The second open source project is a software that was developed until 2007. The project which developed the software was not funded after then. EMSL ELN[ems] was evaluated nevertheless, because it tried to incorporate semantic features. The laboratory notebook was not specific for a certain research field. It included many features that were needed, but cannot preserve data.

Notebookmaker The commercial product Notebookmaker [Nbm] is a very rudimentary notebook. It has simple pages on which the scientist can document his findings. Its design is very similar to a paper notebook. The Notebookmaker is easy to use and has no specialisation. Because of its simplicity it does not support many data formats nor does it integrate tools.

E-Notebook The laboratory notebook, which is most elaborated in the comparison, was the E-Notebook[ENo]. It is a commercial notebook, which specialises on chemical laboratories. It supports many features for chemists, like structures and other chemical software and databases. The E-Notebook can integrate Windows software such as Word and Excel. Due to its specialisation and other characteristics, like platform independence, it can not be used as a general laboratory notebook.

mbllab The mbllab[mll] provided most information on the notebook over an e-mail contact. The notebook is primarily designed to help the project manager of a study. It provides several exporting functions of the results. This notebook is again specialised for chemical use cases.

All in all the analysis showed that especially the support for chemical implementation of notebooks is omnipresent, and more general approaches are rare. The article “A

Review of Electronic Laboratory Notebooks Available in the Market" comes to similar conclusion. The article from 2010 describe some more laboratory implementations and it gives a detailed insight into the current market situation. It concludes that the market evolves from specialisations to generic approaches. With this new trend,

> "every ELN will have its pros and cons when trying to select which solution best fits a customer's needs" [MR10]

This realization makes the right decision for a fitting laboratory notebook for one area very hard. Therefore it is required to find a solution that can be best adjusted to any area. It has to be easy for the user to use but also for the administrator to be managed and customized. Especially if one system is to be used for different disciplines.

3.1.3 Situation of the laboratory notebook in science laboratories

Several examples of real laboratory notebooks can be found "in the wild". They are not always called laboratory notebook, but the main idea is the same: documentation of study results and having a structured approach to research documentation.
Not only the life sciences use laboratory notebooks, but also the New York University recommends its computer science students to use a laboratory notebook (cf. [DJCF]). The usual lab notebook of students are paper based notebooks, with handwritten notes. The paper based approach ceases to be efficient, when starting to work with computerized systems and more elaborated data. A mix of technologies is the most common situation in current laboratories. In [BeLb] the situation was analysed for several Max-Planck-Institutes and the PTB (Physikalische-Technische Bundesanstalt) and it confirmed that a mix of prints, CDs and data links are used to document their work, seldom a is used. A structured and unified approach seemed to be missing. For the DLR the situation is similar, this was verified by doing some interviews with scientists.[8]
When using an electronic notebook, probably the next step is to publish it in the Internet which is defined as "Open Notebook Science" [Wikb]. This movement shares data and results with its community and gets its idea from the Open Source Software. The page [Wikb] provides a list of current laboratory notebooks, which can be freely accessed. On a glance most accessible notebooks belong to researchers from the biological and chemical field, rarely from engineering fields. This could be because a sufficient solution for this field is not provided so far.

3.2 Configuration: Models for provenance and data management

This section explains different models that were developed for the implementation. The design is based on the requirements analysis in chapter 2.1.1 and 3.1.

[8]The question sheet can be found in appendix A

3.2.1 Provenance configuration: questions and models

This section describes the used concept for making the DataFinder "provenance aware". The following use case questions on which the model is based, are concluded on the requirements analysis of the laboratory notebook and developed according to PrIMe[9] "PrIMe is a guided approach for making applications provenance aware"[MMG+06]. The basics of the approach are described in chapter 2. The developed model is based on the meta model as defined in the "Open provenance Model", which is also described in chapter 2. The OPM is the result of the attempt to unify different modeling approaches during provenance development and tries to build a common base to improve interchangeability of models.[OPM] The original PrIMe approach is based on a different provenance model, but to simplify the realization, the differing notation is mapped to OPM annotation.[10]

Use Case Questions

The questions are a collection of questions that need to be answered, when following the GLP as explained in the previous section.
Each question has a short explanation of its application area. This will be extended by a start item (relevant information items), which defines the component that originally holds the information in question. The scope defines all relevant and involved components belonging to this questions. In the processing step the expected return item is defined, as well es the expected steps to get to it.

Areas for Use Cases The use case questions are divided into different areas. Each dealing with various aspects of the data items and laboratory notebook characteristics. These areas are:

Lifecycle Actions evolving around the life cycle of a data item, such as adding, editing, changing and deleting are targeted with this set of questions.

Origin The history of a data item, especially timely and logical successors and predecessors of an data item are subject of the Origin questions.

Quality assurance In this set of questions dependencies to standard procedures, study plan and other reports evolving around specifications for the generating data items are targeted.

Credibility Relevant questions belonging to the authenticity of a data item, that are realized by signatures and archiving mechanisms, are aggregated in the credibility set.

For each area two representative questions were selected and printed here. A full version of all questions can be found in appendix A.

[9] PRovenance Incorporating Methodology.
[10] agent == actor

Questions - Life cycle This set of questions evolves around the actions that happen with data item, the data and the meta data. These actions can be either adding, changing or deleting. Each action can be performed by hand or automatically via a software. Different handling mechanism such as deleting, changing, adding, downloading and adding are not further distinct.

- **When was data item X handled?**(LC1)

  Application description Aim of this question is to identify the changing process of the data item.

  Relevant information items (Start item) data item X

  Involved components (Scope) development of a data item from now, back to its first appearance in the system

  Processing steps Return a history with changing dates

- **Who handled most data items in experiment X? (LC2)**

  Application description To know who was most active during an experiment can be helpful for interpretation or familiarization with an experiment.

  Relevant information items (Start item) actors, experiment X

  Involved components (Scope) all data items belonging to experiment X

  Processing steps Return a list of actors and the belonging aggregated actions for experiment X

Questions - Origin This use case question evolves around the questions around the matter of predecessors and successors. They are either influencing or influenced by the data item in question. It asks for concrete time bars and chain of events.

- **What data item was handled timely before data item X was added? (O1)**

  Application description This is an answer important to know, if one needs a chain of events leading to a data item.

  Relevant information items (Start item) data item X, (experiment)

  Involved components (Scope) from data item X's time stamp, to the first entry

  Processing steps The system is searched for all actions happening within in a certain time period and then ordered over time. A time bar (sequence) of documents that were altered before data item X is returned.

- **What is the logical successor of the data item? (the visual data gained from the raw data) (04)**

  Application description This use case question evaluates the influences of one data item on following items, with the focus on logical dependencies.

Relevant information items (Start item) data item X

Involved components (Scope) from this data item, to the data item, that has no successors anymore

Processing steps Get the data item X and evaluate its successors based on relations to the data item. Return a list of data items, their timestamps and a corresponding action.

Questions - Quality assurance This set of questions aims at understanding the requirements leading to a certain data item. It evolves around study plans, defining the experiments environment, and standard procedures, that define general usage of tools and experimenting devices.

- **To what standard procedure/experiment/study plan belongs the data item X? (QA1)**

  Application description This use case is to know what are the influencing documents for one data item.

  Relevant information items (Start item) data item X

  Involved components (Scope) from data item X, to standard procedure/experiment/study plan.

  Processing steps Return the data item representing the procedure/experiment/study plan.

- **What data items belong to a report X? (QA2)**

  Application description For credible reports it is useful, to have the data items, that helped creating the report. They need to be accessible for other scientists, trying to reproduce results from the report.

  Relevant information items (Start item) report X

  Involved components (Scope) from report, to all data items belonging to the report or experiment

  Processing steps Return a list with data items that are part of the report

Questions - Credibility The following use case questions evolve around the credibility of data. Credibility can be for example produced by signing the content of the data item with a signature. The requirements on "Genuineness" and "Verifying results" in chapter 3.1 give more information about the necessity of credibility in laboratory notebooks.

- **Who is responsible for data item X? (C1)**

Application description This questions aims at the person who takes responsibility for the correctness of the data. Most often it is the person who signs with his signature the data to be valid, and especially the one who first signed it.

Relevant information items (Start item) data item X, actor

Involved components (Scope) from data item X, to first version of the data item with signature

Processing steps Find all versions of the data item, and then return actor, who first signed the content.

- **How many verify data item X to be valid? (C2)**

  Application description This question aims at the grade of credibility for a data item. Depending on the amount and quality of a signature, it can be characterized. Also if there exists a signature for each state of a data item, credibility is consistent.

  Relevant information items (Start item) data item X, signatures

  Involved components (Scope) from data item X, to its earliest version with a signature

  Processing steps Search for all versions of the data with a signature and then aggregate over each signature. It results in a list with signatures.

Actor and Process Identification

After analysing use cases the next step in PrIMe is actor identification. In this application of the PrIMe approach, different processes in conducting a laboratory notebook dictate the workflow. Each process is controlled by a different set of actors. These sub processes are preparation, execution, evaluation, interpretation and archiving. The actors are either a person, a software or an instrument. During each process several artifacts are generated.

Developing the Open Provenance Model

In the following section each sub process is further explained and an OPM model is proposed. A model for the whole process sums up this section.

Preparation Process The preparation process is defined as the phase where the scientist searches for the correct manuals, gets familiar with the common procedures and defines a study specific plan. It is the first process to take place in each study and should be entered first into a laboratory notebook. Further preparation processes can be part of a study, for example when more detail is needed for the design of an experiment.

input Manuals, Standard Procedures, Equipment information

output study plan

The corresponding OPM model can be seen in figure 3.1:

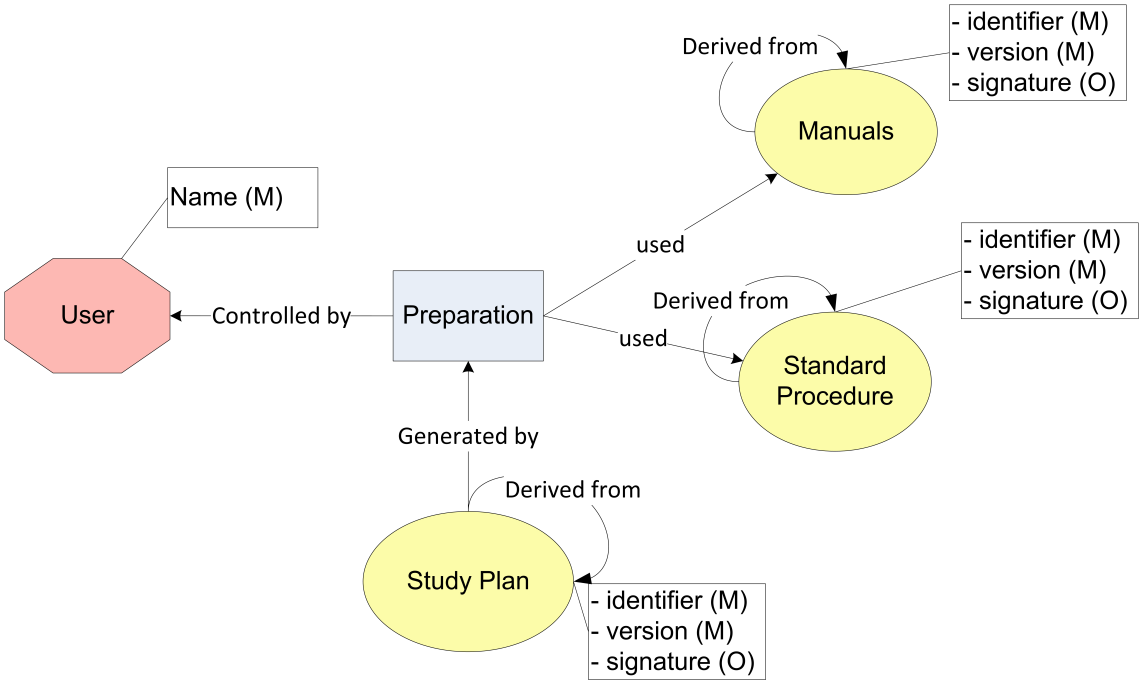


Figure 3.1: OPM model for the preparation process

The experimenter controls the preparation process and actually uses the different documents in order to generate the study plan.

Execution During the execution phase the study plan is the main document on which the experimenter relies his work. The experimenter extracts from the study plan an individual experiment plan. The experiment plan gives information about the individual experiments. During this phase all individual experiments produce experimental data, calibration data and interpretation data. The results belong to an experiment and a specific study plan.

input study plan

output experimental data, calibration data, interpretation notes

The following model in figure 3.2 presents the OPM procedure.

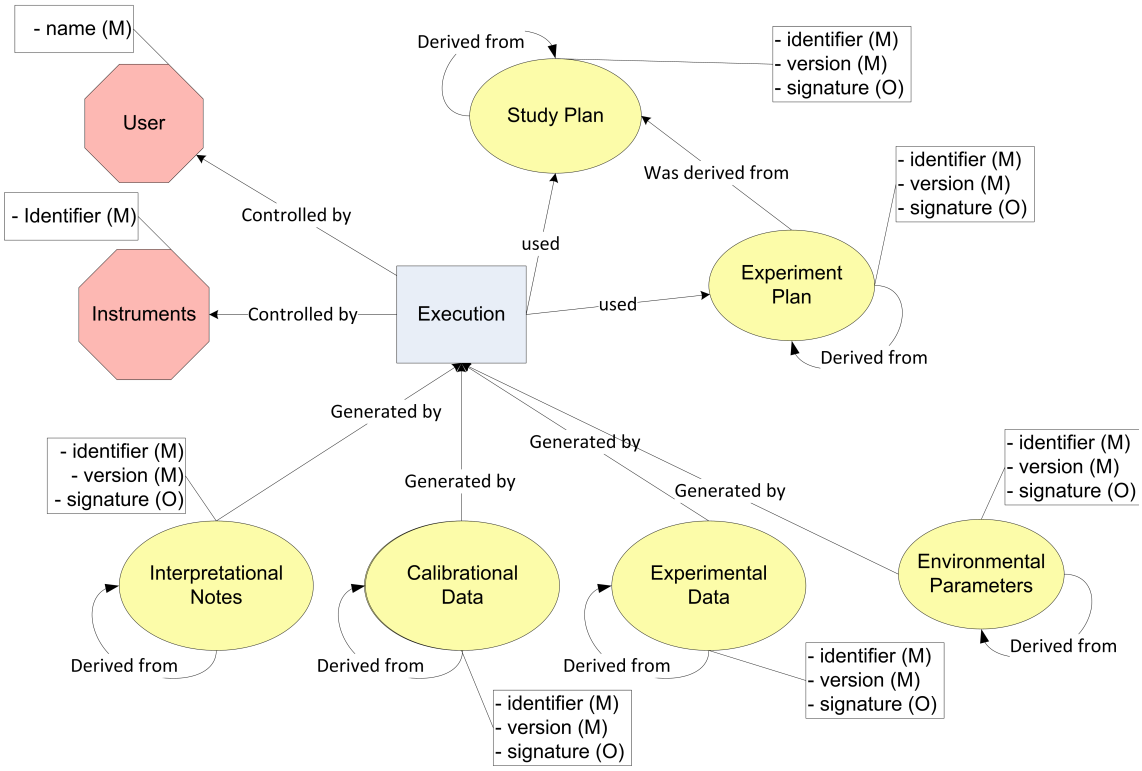


Figure 3.2: OPM model for the execution process

In this phase users and instruments control the process. An instrument can be an executor, if an experiment is performed automatically. The study plan or experiment plan is needed as input for the process and the process generates several unprocessed data items, such as experimental data and calibration parameters.

Evaluation For evaluation experimental data and calibration data is used to generate results. The items can be processed with either simple calculations applied to the experimental data, up to numerical analysis and visualization with other tools.

input diverse raw experimental data, calibration data, user

output evaluated data

Figure 3.3 shows the OPM model for the evaluation process.

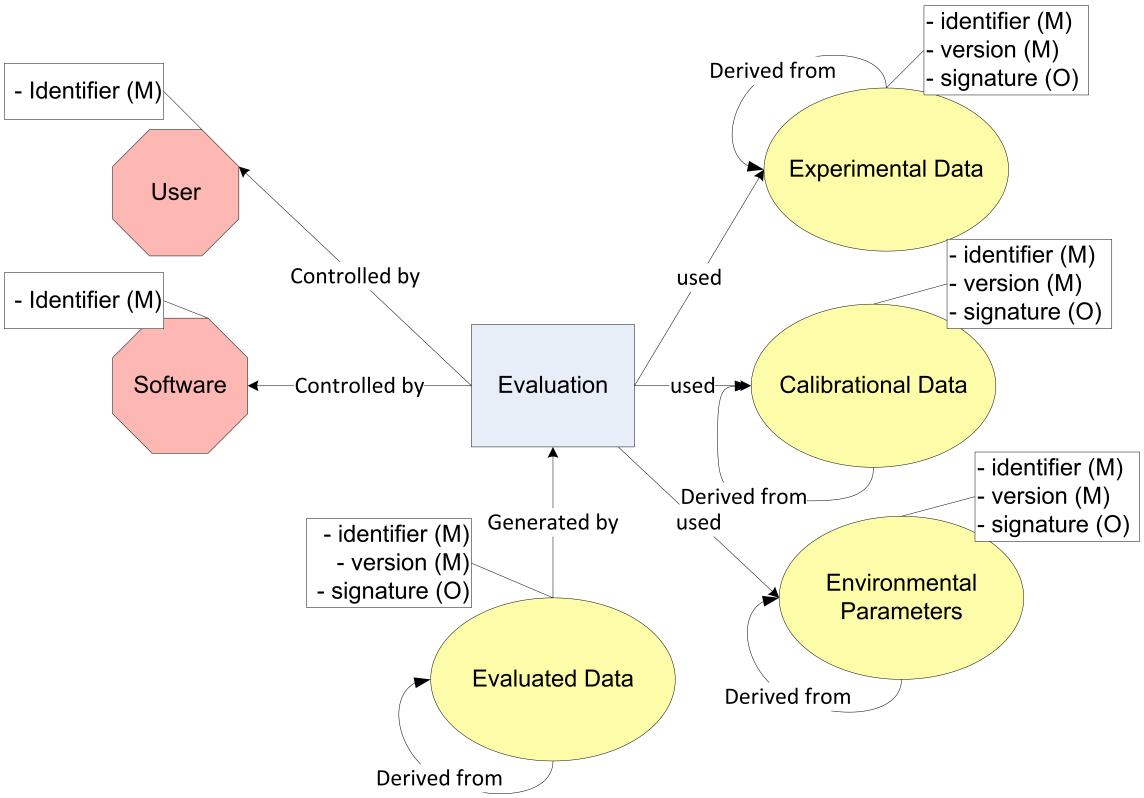


Figure 3.3: OPM model for the evaluation process

The process is controlled by the user or a specific software designed for evaluation of certain data. The data that is generated in this process usually consists of data adjusted to calibration data or external parameters. Visualization of experimental data can be a generated output as well.

Interpretation The interpretation process follows the evaluation and interprets the visualized and adjusted data. The interpretation often results in a report or in other interpreted data, that emphasize the gained information of the study or experiment.

input notes, evaluated data, validated data

output interpretation data, study report

Figure 3.4 shows the corresponding model in OPM notation.

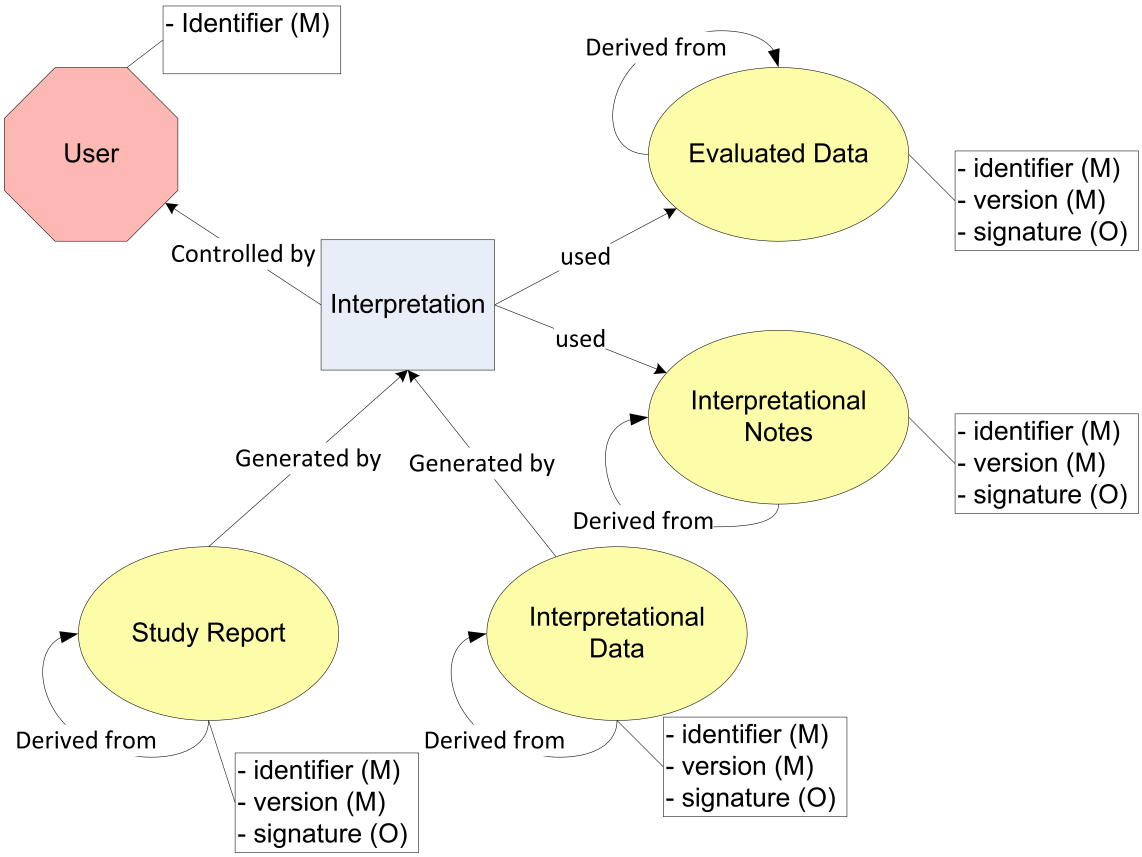


Figure 3.4: OPM model for the interpretation process

This process is controlled by the experimenter of the study. As input, notes,that were made during the study like fleeting observations, and evaluated data are taken. The output is usually a study report or other data, which is needed as input for further experiments.

Archiving The last sub process in a scientific documentation workflow is the preservation of data items. Everything that has to do with the study needs to be archived and kept accessible for a time span. As input manuals, standard procedures, all generated data, all evaluated data, each note and each scrap of paper are taken.

input all relevant data items

output archiving

The model in figure 3.5 represents the OPM model of the process.

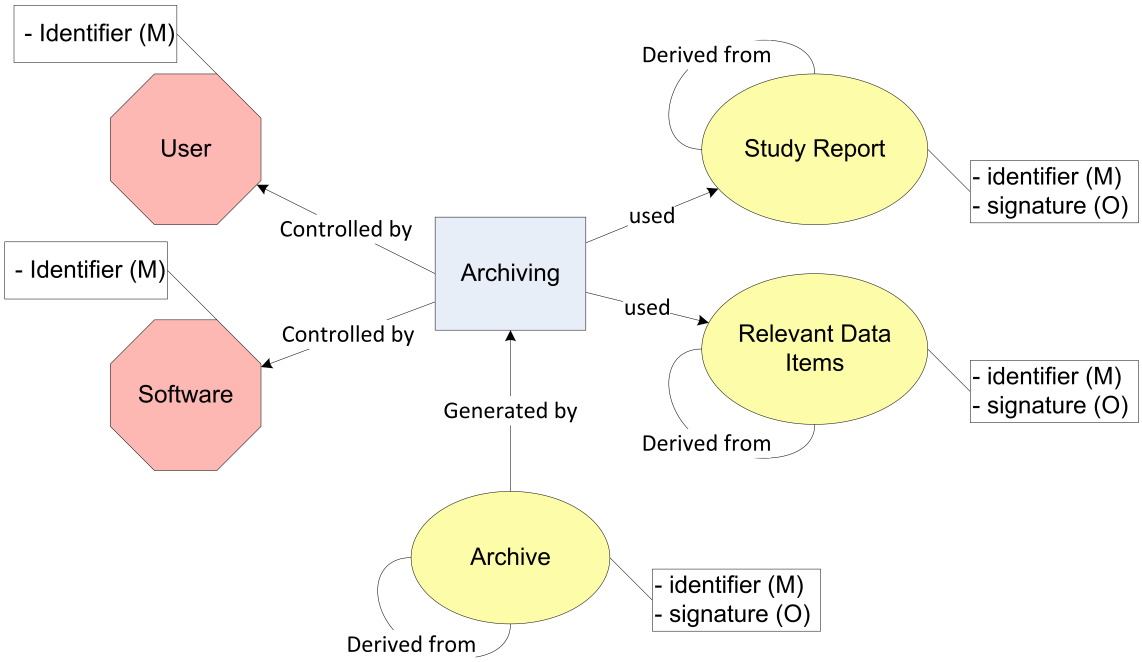


Figure 3.5: OPM model for the archiving process

This process can be controlled by an user, who archives the data, or a software mechanism is developed, that automatically archives all data. The archive is the final result of a study.

Laboratory notebook model In figure 3.6 all previous processes are combined and the model of the complete workflow is presented.

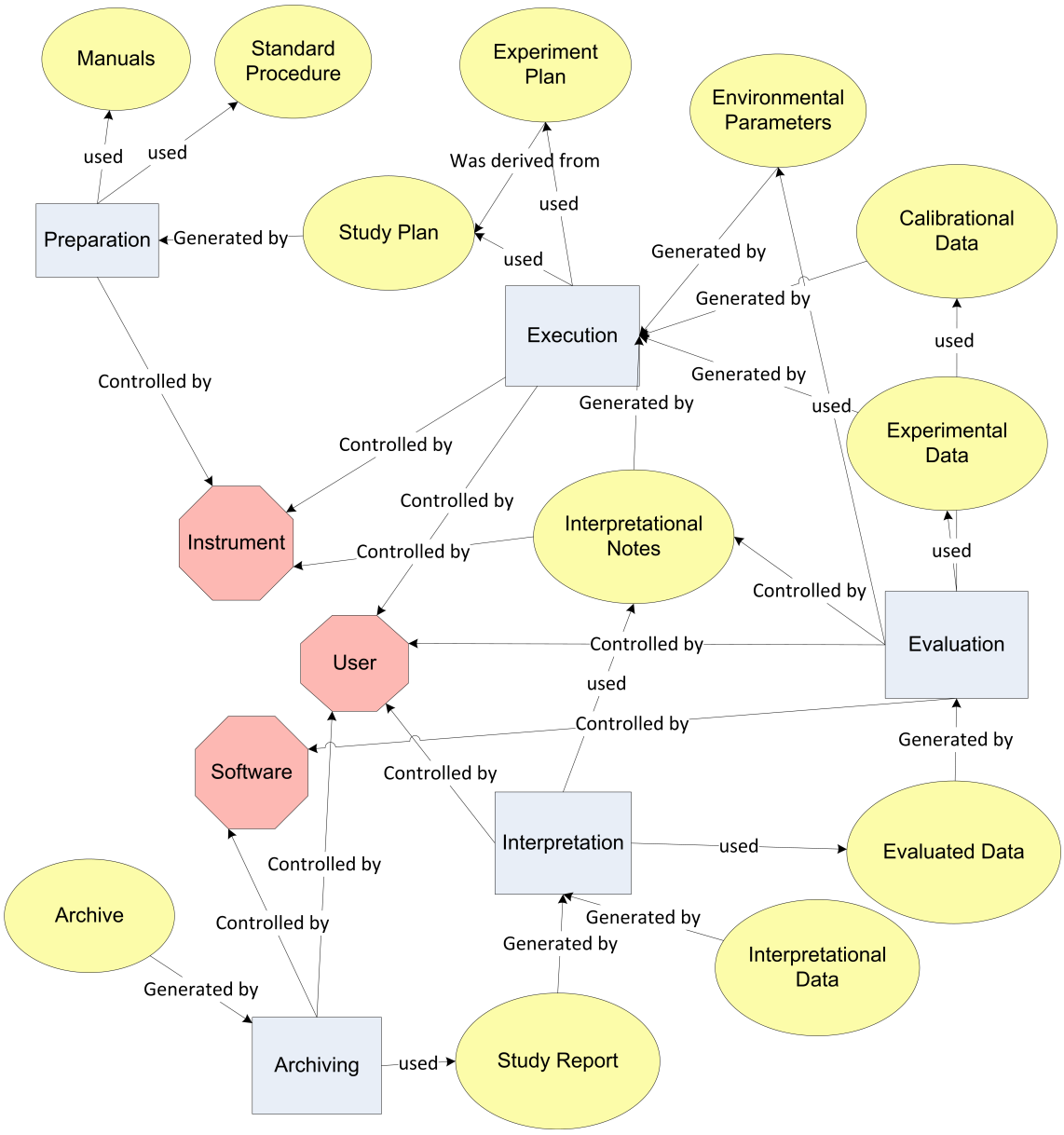


Figure 3.6: OPM model for the whole process

The figure shows that all processes have similar actors. They are a main component of the model. Also the data items experience a life cycle which are defined by the processes and which is similar to the life cycle from figure 2.4.

3.2.2 DataFinder configuration models

This section describes the configuration of the DataFinder. The general configuration consists of two models that need to be defined. On the one hand a data model, that describes the structure of the managed data and its necessary meta data. On the other

hand a model of a storage system is configured. The data store configuration remains as the default configuration. The default configuration uses for the meta data and the meta data the same server. So this chapter will describe the development of the data model.
Similar to section 2.1.1, where the data items are divided into two parts, the data model for the DataFinder can be divided and later assembled. The definition of the model is mainly based on the conduct of a study defined in [OEC97] and the needed documents for good laboratory practice in this document.
The general data model and the study specific data model are described in the following separate sections.

Data model for general data

The data model for the general data sorts the data belonging to apparatus, test systems and test items. A schematic diagram can be seen in figure 3.9.

Figure 3.7: General data model for the DataFinder

The figure shows, that within this part of the DataFinder general information of the departments apparatus are managed. Each apparatus has some sort of calibration records or handling procedures included. All information provided in this section is for the general use of the scientist and gives an overview of the inventory of the department.

Data model for study specific data

Since each study is unique, the data model needs to be as generic as possible. The model as in figure 3.8 shows this approach. It is modeled against the analysed workflow of conducting a study from chapter 2.1.1. The figure 2.2 shows the model.

Figure 3.8: Study specific data model for the DataFinder

The model sorts data items to the specific phases. For example when preparing a study, it is common to use a manual to read instruction on how to use an apparatus. All relevant information needed for this can be collected or referenced there.
The experiment collection was included, because to one study belong several experiments with different aims and results.
Referencing between documents from different phases is possible and wished for.

Representation of the models in the DataFinder

When integrating the models from the previous two sections into the DataFinder, a possible repository shows figure 3.9.

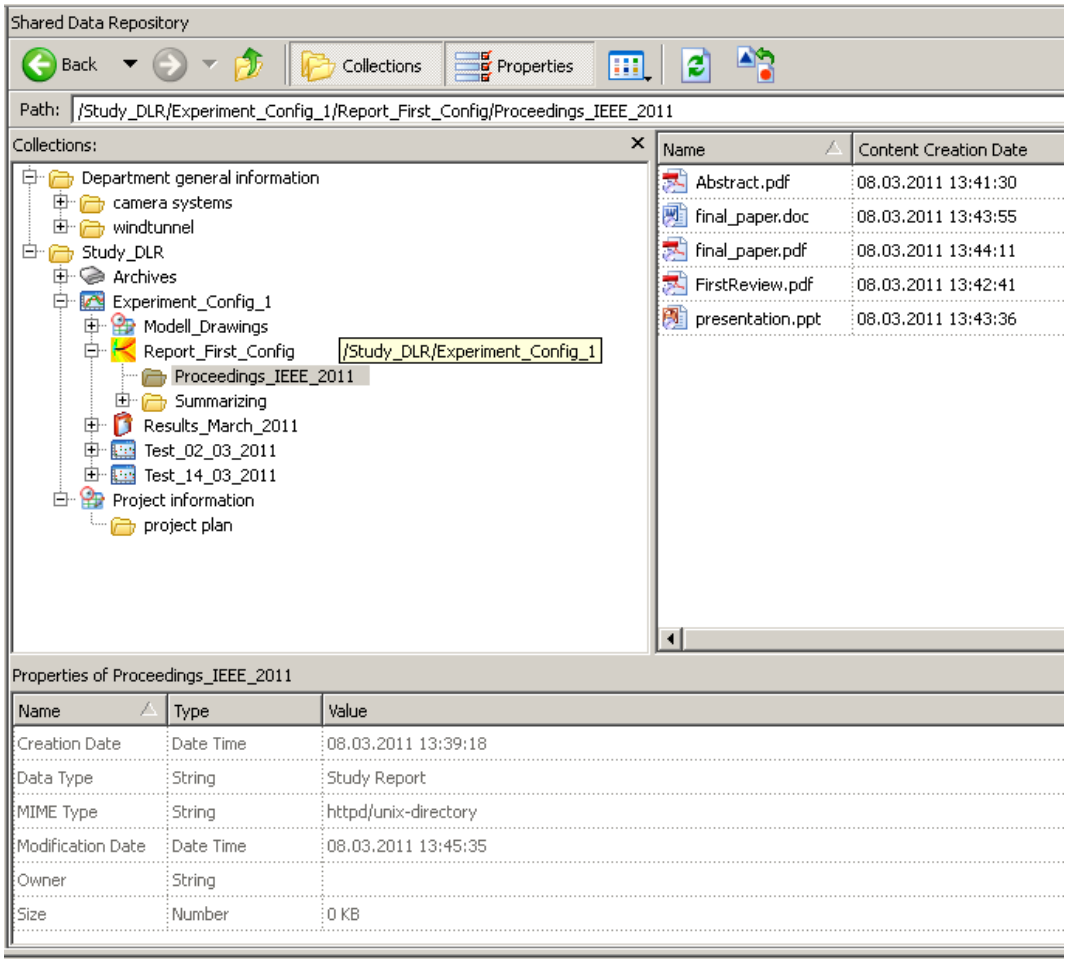


Figure 3.9: Data model integrated into the DataFinder

The names from the model are not visible. The types of the folder, which are the definition of the model can be found in the properties section in the lower part of the screenshot. The icons that can be seen in front of some icons are defined in the data model as well. They identify a certain data type intuitively in the user interface.

3.3 Implementation of Requirements: Chain of Events, Durability and Credibility.

After having analysed the requirements, and the DataFinders current capabilities [11]. Three main characteristics are targeted in the following implementation:

- "Chain of Events" with a provenance integration

[11] a summary is in chapter 4

- Secure “Preservation” with incorporating a web service to handle it
- “Credibility” with enabling signatures of items

The implementation might target several requirement, but the chosen ones are the most important features, on which to focus.
First a mechanism to support “chain of events” and its integration into the DataFinder is shown. Then a concept of the integration of evidential preservation into the system is described. And at last a concept to provide credibility to data items is presented and its implementation investigated.
Each part is implemented with a different agile development approach:

Chain of Events is implemented with the concept of prototyping. In literature a prototype is defined as: “A prototype is an initial version of a software system, which is used to demonstrate concepts, test designs and to understand the problem and its solutions.” (adapted from [Som07] p. 443) The resulting prototyping process uses this first prototype, takes relevant parts and designs a next prototype, until finally it has matured to a final version.
This development process is chosen, because the implementation of the feature is rather complex and needs requirement analysis and usability tests. Also in the beginning a concrete implementation concept could not be defined.

Preservation is implemented as test driven development.“Test-driven development is a discipline of design and programming where every line of new code is written in response to a test the programmer writes just before coding.”[JM07]
The implementation of the archiving scripts is implemented as the Jeffries and Melnik describe it. First a test is designed and then the feature implemented. This approach is used, because the required information of the project is still in progress. Changed implementation of the projects web service can then be tested easily, and adjusted to new requirements.

Credibility is intended to be implemented with a pair programming approach. Pair programming is, when one person(“driver”) implements a feature, while another person (“observer”) watches and corrects him. The roles can be switched. This approach was chosen, because the developed concept should be integrated into the core of the DataFinder. To make such a change the involvement of more responsible persons is needed.

3.3.1 Chain of Events: Provenance integration into DataFinder

This section describes the integration of a “chain of events”. For the realization provenance is used. First a previously developed provenance storing system is evaluated and the necessary adjustments described.Afterwards the integration into the data management system DataFinder is discussed.

Adapting the provenance system: noblivious

For the DataFinder data management system to be provenance aware, the model that was developed in section 3.2.1 needs to be integrated into a provenance system. The basic software system, that is used is designed for recording a software engineering process and needs adaption.
To use it in the context of the laboratory notebook it needs to be able to store the prior developed provenance model and provide the correct interfaces for storing and querying. When trying to adapt the software engineering system to a lab notebook system several issues occurred. Then rather implementing another specific system, a general system was designed.
The next sections explain the process of adapting a Software Engineering provenance system to a general provenance system. First the system is evaluated. The evaluation shows the problems that occurred, when trying to adapt it to the laboratory notebook specific model. Methods to fix the issues are described. In the end the resulting system is presented.

Laboratory notebook specific provenance system The original design of the "noblivious" system has for each defined process a distinct REST interface. For each artifact and actor within the model a distinct creation method, with individual identifiers and types were defined.
So the first steps for implementing the lab notebook noblivious system was to add the REST interfaces for the different processes. As well as adding different identifiers and methods for each artifact and actor.
When doing so difficulties occurred, such as redundant code and difficult handling of the service. The difficulties are listed below and explained.

Redundant attributes Each artifact has attributes, such as identifier and signature. According to the original implementation of the "noblivious" - system, same attributes of different artifacts (data items) need to be named different. This results in a lot of variables for similar information.
In order for this design to be obsolete, there must be a guarantee that artifacts in spite of same attribute names can be identified. This can be realized, if unique values are added to the attributes. If the unique values are not possible, the queries must be designed more specific.

Redundant create methods For each different type of artifact, such as manual and standard procedure, within the provenance service a different method needed to be implemented. The methods only differed in the attribute names they assigned to the nodes.
As soon as the different attribute names were unified the different create methods were obsolete. They are standardized into a single "create-Node" method.

Unified handling methods In each process interface the given artifacts are handled separately. The difference of the artifacts were the relationships that were assigned between them. So they the procedure was only different, if an artifact between inputs and outputs.
This is unified for each process, so that each interface uses for generation of their artifact global input, output, actor and process methods.

Version attribute For different versions of the same artifact within the original "noblivious" system one needed to add a specific process. This seemed redundant since the OPM defines a relationship for this use case: "derived from".
To implement this the attributes of the artifact are extended with a version.

Multiple input and actors The output of a process can be influenced by more than one actor or more than one input of the same type. This scenario was not implemented within the original "noblivious" system.
Therefore the parameter for one artifact is changed into accepting a list of artifacts. The artifacts are separated by a delimiter. The change makes it possible to generate several nodes of the same type for one process.

Unified REST interface A difficulty in adjusting and using the system was to integrate consistent names of the parameters for the REST service. Each process belonged to one REST interface and in each request the artifacts was defined by a single parameter. In order to have a flexible provenance system and no dependencies on the names of the parameters, the interfaces need to be unified.
A REST interface was developed, that only differentiated between input, output and actor instead of a separate parameter for each artifact. This makes it necessary for each parameter to have an additional artifact type information. Another delimiter for this information within the identifier was specified.

The above described difficulties and the proposed solutions resulted in a more common implementation of the service. As a result a lab notebook specific provenance system was obsolete and the more general provenance system was used. The result was successfully JUnit and system tested and is presented here.

General provenance system The resulting general "noblivious" provenance system is implemented in Java. It has a graph database as backend, a interface as frontend, a gremlin servlet to access the database and a new REST interface for Gremlin queries to the database. Figure 3.10 shows how each of the component is integrated into the complete architecture.
By removing use case specific information from the inner processes, the internal code is more modular and general usable.
The new general REST interface for storing provenance information accepts strings corresponding to a specific format. One process at a time can be handled. For this

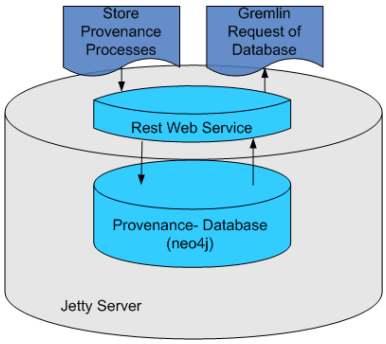


Figure 3.10: Design of the provenance system's architecture

process several inputs of different types, several actors and one output can be provided. The service can handle artifacts of the same Id in different versions.
The advantage of this implementation over a use case specific implementation is that: if the provenance model is changed, there is no need to touch the service implementation. Only the applications filling the system via the REST interface need to be aware of the change and adjust their requests to it. A disadvantage is, that types of the elements need to be named consistently throughout the usage of the system. Consistent naming is necessary in order for the system to deliver consistent and reliable results, when queried.
The general provenance system can be queried, by the original web servlet, or via the added REST interface. The interfaces accept Gremlin queries.[12]

Adjusting DataFinder

The information filling the provenance database is delivered by the DataFinder, and is send as REST request to the system. This section explains the implementation of it.
The development was done as prototypes, and with each step a slightly different version was implemented. All implementations use the script extension of the DataFinder. If changes within the DataFinder core were necessary, they are explicitly mentioned and were implemented in correspondence with the DataFinder development team. For configuring the DataFinder the data model from section 3.2.2 is used.
In the following the different approaches, its advantages and disadvantages are discussed. Also influences on other parts of the implementation, e.g. the data model, are mentioned and explained.

First implementation: Manually selecting files for a process In the beginning a straight forward approach was selected:

[12]For all provenance use case questions, that were developed in section3.2.1 the gremlin queries are provided in appendix A

Aim is to select a single file and send it to the provenance system to be stored in the database.

Realization when having selected a file within the DataFinder repository, the context menu offers the option to store this in the provenance service

Issues Several Problems occur, such as:

- selecting files from different folders is not possible; and files from different folders are needed for one process
- the user needs to remember all the input and output files needed for one process and therefore be aware of the correct provenance model
- only possible to send one input type at a time

Impact on other components On the provenance system: having to be able to send several inputs of the same type at the same time

Second Implementation: Selecting process to send information The second implementation tried to eliminate problems of the first implementation such as: selecting files from different folders.

Aim is to select a process within the repository and send its content to the provenance system.

Realization is: After selecting a process, the script extracts from the process folder its process type. Then gets the children of the process and sets the documents within the children as input and output for the process.

Issues that occurred were:

- Separating between input and output files; It was not possible, since not in all process folders have all necessary input and output files included
- If there were several output files it is not possible to extract the input file for a specific output
- The user needs to remember the execution of the script. So it might occur, that processes are not stored in the order of occurrence.

Impact on other components The DataFinder data model needs adjustment such as adding the information of input and output types to each file.

Third Implementation: Constantly checking for file imports The third implementation approach tried to solve the issue of the user needing to think of the provenance service.

Aim is to automatically send provenance information about a imported files. An import usually means a process took place.

Realization The script needs to be executed once at the beginning of each DataFinder session. It constantly checks on which files and processes locks occur, extracts necessary information of the file and then sends the information to the provenance system.

Issues that occurred were:

- too often items were locked that were not new or interesting for the provenance of an object
- too many threading issues occurred when constantly checking for changed objects
- the implementation created a lot of processes that only had one item each.

Impact on other components The data model needs concrete information on the correct input files for a specific output.

Before the fourth implementation is described a short insight on the changes within the DataFinder is given:
In order for the provenance script extension to work automatically it needed to be notified by changes in the repository. To enable this an listener mechanism is added to the DataFinder. The DataFinder is now able to register listeners for certain events. New events can be easily added and in the script it is possible to register for the same events.

Fourth Implementation: Execution on file import The fourth implementation uses one of the before described event listener and has the following characteristics

Aim is that after an import of a necessary process information is extracted. Also the correct input files are automatically extracted and send to the provenance service.

Realization The script needs to be executed once for each session, which then registers the script as listener for an import event. On a file import, the script extract the necessary information and calls the provenance service.

Issues are that not immediately correct inputs are chosen. Instead a list of files that were imported before is sent to the service.

Impact on other components This script impacted the DataFinder and its script API by making a event mechanism necessary.

Fifth Implementation: Execution on file import with a dialog This implementation tries to address the problem of choosing the correct input file.

Aim is that the user gets to select the corresponding input files for an output file via an dialog. The output file is a file, that has been imported recently.

Realization is similar to the fourth implementation, but a dialog with a possible input list is opened. The user can then choose multiple input files, that are needed for the provenance service. Figure 4.1 in chapter 4 shows the dialog.

Issues Some problems that occurred were:

- how to work with changed documents of the same identifier
- thinking of registering the listeners at the beginning of each session
- the pop up dialog might be bothering, especially on importing several inputs.
- increasing number of files, increasing number of inputs to chose from

Impact on other components Opening the dialog resulted in GUI threading issues. To avoid it the script API is extended with a mechanism to access GUI threads of the main application.

Sixth Implementation: Versioning and automatic execution In this implementation the issues of automatic execution and the handling a file with different versions was addressed. Also the script was adapted to the latest version of the provenance service, as described in section 3.3.1.

Aim is to enable automatic recording of the provenance service. This forces the usage of the provenance service. Also it was updated to the latest service implementation, for example adding a version to the data items.

Realization In the DataFinder each script extension is processed on import. The DataFinder looks for certain tags to analyse the script. A tag could be a definition of data types the script is used for. So for the realisation of the automatic execution on import a new tag was added to the DataFinder script processing unit. As version the meta data information of the "modificationDate" was chosen. The "modificationDate" seemed suitable, because it changes with each edition of data or meta data. The sending process was adjusted in the service component of the script.

Issues This for the time being final implementation gives a solution for the provenance service, without having to think about the execution of the script and only having a rudimentary knowledge of the provenance model. For proving that the implementation is possible this status is enough and can be used successfully. Still remaining issues are for example:

- a bothering pop up dialog
- increasing number of files to chose from in the dialog

Impact on other components To the script registering component within the DataFinder core a new script processing tag was added.

3.3.2 Preservation: storing data evidentially and long term

Scientific results need to be available for a long time. That way other scientists can reprocess it in the future or evaluate it under different aspects. To make sure, that the data is valid, it needs to be securely stored with a evidential process.
With paper based notebooks it is easy to realize: Write the data in a notebook, put a date on it, sign it and store it in a locker.
For a digital notebook this process needs to be further conceptualized and established, and therefore the BeLab project was started. As one of the aims it develops a web based service to support the storing procedure. Before the web service was integrated into the DataFinder an internal study evaluated its capabilities of long term preservation according to the standard OAIS. The results of the study are shortly described and a reason for the integration of the BeLab service given. The BeLab web service and its interface will be shortly described in the following section. Afterwards the integration of the service into the DataFinder infrastructure is explained. This integration is based on Python scripts and depends on the provenance service, which is described in section 3.3.1. [13]

Capabilities of the DataFinder according to OAIS

In a DLR internal study[Datb] the DataFinder was evaluated for its capabilities to work as an OAIS.[14]. The result of the study is that functionality such as accessing inserted data, storing data and meta data as well as management of the data is possible. Some aspects such as incomplete implementation of the OAIS information model and migration functionality are missing. They can be fulfilled if the tasks are done by management personell. To be used as a fully qualified OAIS a few characteristics like persistent Ids or DOIs need to be integrated. The functionality of providing persistent Ids and also evidential features was not implemented. To use a service that can provide the archival features was decided to be a good alternative.

BeLab WebService

The BeLab team currently develops a concept and a prototype for storing data evidential secure. The concept deals with storing evidential according to German law and the highest protection possible. It focuses on the usage of digital signatures in order to make digital data more reliable and valuable.
The implementation is a web service using several standards. Technologies that are used, are WS-security and standardized archives.
WS- security is a technology with which transporting data with security information is possible. Security information are passwords, certificates and a secure transport protocol. Generally it uses a SOAP message to transport the data and in the header,

[13]The results of the solution can still be used separately without a provenance integration.
[14]The OAIS is described in chapter2.3.1

authorization information is provided.
For the data body, different data types are allowed: archives and single files. In a first implementation any archive is accepted, but in the end some sort of standardized archives are used.
When accepted by the system, the archive or file is processed and evaluated. The evaluation states up to which grade the data is reliable to endure the time span and a court hearing. In the end the data item is stored on the service's storage location. As response the permanent id of the item is sent to the user.

Integration of the service to the DataFinder

The service is integrated with the agile concept of "Test Driven Development". The main idea of the concept is to write a test for a feature one wishes the application should have. Then after the test failed, to implement the feature, so that the test stops failing. The process starts over, until there are no features left. Test Driven Development was chosen for this script, because the implementation will need adjustments with new prototypes of the service. The requests and responses to the different services can then easily be tested.
For the script to archive data from the DataFinder into the BeLab Web Service, the following features were chosen and implemented:

- choosing elements valuable for preservation, e.g. all data belonging to a study report
- Packing data items into an archive
- Calling the web service and sending the archive
- Handling the response of the web service

Each feature, their tests and the implementation will be described in the following subsections. For each feature several tests were defined. First the test is discussed. The aim that is supposed to be reached by this test is mentioned. Then a short description of the resulting implementation is provided. In some cases issues are explained.

Choosing elements worthy of archiving Not all data is usually archived. When choosing elements that are worthy of keeping, only a subset of all available data is kept.
To target this concern, the script implements a strategy to extract a few relevant items, which is based on the implementation of the provenance model in section 3.3.1. The user is supposed to chose a study report which he wants to store securely, the script then extracts all data, that belongs to the study report, in each state. This use case is described as question no. QA1 in section 3.2.1.
So task of this part of the script is to connect to the provenance service. Post a query

to the provenance service. Then get a response with ids of all relevant data items belonging to the study report. These ids are used in the next part, when the items are extracted from the DataFinder and packed to an archive.

Test **Testing the connection to the provenance service** (testInitializingConnection)

Aim Test the right configuration and the availability of the service

Implementation Establish a connection to the service including authorization information.

Issues In order to have a service that can be tested easily, the provenance service was extended by an interface which automatically generates a database which has the needed structure. This test database is specific and not integrated into the general service.

Test **Testing to send a simple request**

Aim Figuring out how a request needs to be stated

Implementation The service needs to be requested in several steps. A query usually consists of several sub queries. For each sub query, a new request is sent to the service.

Test **Testing to send a study report request**

Aim Aim of this test case is to generate a more complex request and to get the correct result.

Implementation A method to generate queries is added to the class. In this method a study report query is implemented. The query is the answer to the provenance question. The identifiers of the items belonging to the study report are returned to the next part.

A screenshot of using the script extension in the DataFinder can be seen in figure 3.11. It shows how the script extension is chosen.

Extracting data items from the DataFinder and packing an archive The next feature is implemented, is the interaction of the script with the DataFinder. The content of relevant data items need to be extracted from the management system and then packed to a compatible archive.

Test **Testing to extract a list of items from the DataFinder repository**

Aim Process of extracting the content of the items from the previous section is tested.

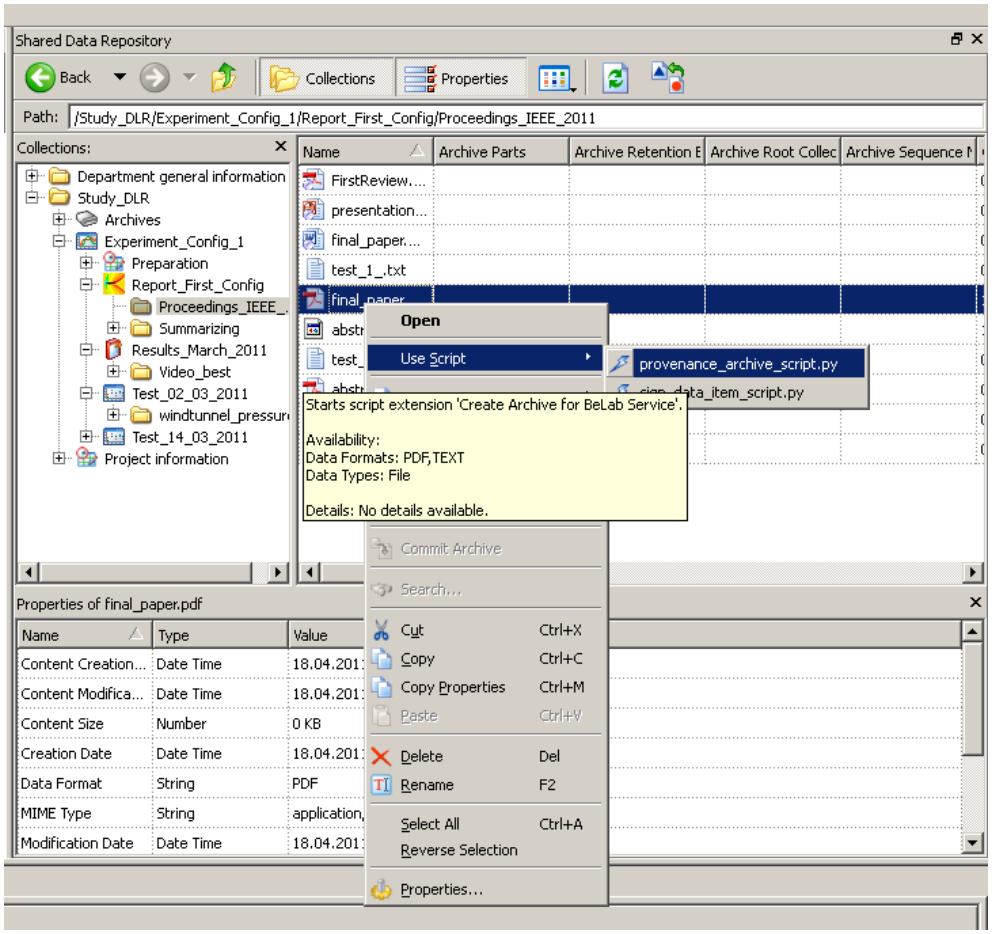


Figure 3.11: Calling the script which internally calls the provenance service

Implementation The ids are looked up in the DataFinder repository. Then the corresponding file objects with the content of the ids are saved in a Python object.

Test Testing to extract signatures for corresponding items

Aim Process of extracting signatures belonging to the items from the previous section is tested.

Implementation The ids, with an altered file extension, are looked up in the DataFinder repository. The content is extracted and returned to be added to a temporary Python object.

Test Testing to pack a simple archive

Aim The initial version of the service accepted simple archives. So it is tested, if this is generated.

Implementation Getting the file objects of the items, and adding them to a tar archive.

Processing the call to BeLab web service For successfully archiving in the service, a connection needs to be established. The archive sent and the response processed.

Test Testing to connect to the service

Aim To test, that the SOAP service is correctly established and the correct header set.

Implementation Several SOAP libraries were tested. In the end "suds"[sud] was chosen, because of its easy handling and support for ws-security. The wsdl is loaded and then a client, which is used throughout the storing process, is generated. The client has the security information added to the header.

Issues The prototypical service implementation only required, password and user name authorization. An implementation with certificates was not tested.

Test Testing to send an archive

Aim Using the beforehand configured client to send the data to the service.

Implementation The wsdl was loaded before, so the function of the service could directly be accessed through the client.

Issues Currently not a real id is returned, but a string. So the string is further processed.

Test Testing to set archive ids

Aim To test the setting the id from the service into the DataFinder repository. It completes the integration of the service, so that later the item is found again.

Implementation First the Archive that was created was stored in the repository. After the transaction of storing it in the archiving service was successful, the response was evaluated. Then a new property created and the id stored.

3.3.3 Credibility: Integration concepts for digital signatures

To authenticate and validate a certain data item, signatures are used. When having a paper based notebook, signing a page is easy. If an electronic laboratory notebook is used, more difficult. Especially if the provided signature has to be evidential.
This chapter describes the concept of adding digital signatures to the data management system DataFinder. The concept focuses on adding key based signatures and certificate based signatures.
Since there are differences in how to sign data items and meta data items, the matters are handled separately. Before each concept is described an introduction to digital signatures is given.

General digital signing procedure

For establishing a digital signature cryptographic scheme is needed. Those are concepts to ensure digital signing to be secure. A scheme is defined as the following:

> "Digital signature schemes allow a signer *signer S* who has established a public key *pk* to "sign" a message in such a way that any other party who knows *pk* (and knows that this public key was established by *S*) can *verify* that the message originated form *signer S* and has not been modified in any way."(cf. [JK08] p.421)

In other terms a scheme can realize electronically, what a signature does on paper. The research field on cryptography proposes different schemes for signing. They are examined thoroughly on security issues. The implementation is based on mathematical definitions, theorems and assumptions.
Except of in a "Random Oracle Model" no truly secure signature scheme has been found (cf. [JK08] p.426). Most of the schemes are vulnerable, because no truly random functions exist to generate secure keys and signatures. This is the reason the model was introduced. The Random Oracle Model makes the schemes mathematical secure but practically still vulnerable. This means further research needs to be done, and the practical implementations of the schemes improved. Further they need to be adapted to current calculating abilities.
The (currently) most secure way of signing a data underlies the "Hash and Sign" Paradigm (cf. [JK08] p.429). The paradigm means: Generate a hash of your item first, then sign it with a common signing algorithm. To be secure the hashing algorithm needs to be collision resistant[15] and the signing algorithm not forgeable. A common implementation for this is SHA for hashing and RSA for signing. To issue a signature either a public key is used or a certificate. The credibility of the signature is based on the key used for it, the more credible the issuer of the certificate or key. The more credible the signature and its usage in front of court.

Concept for signing data items

To sign data items, two concepts are propose: One is to sign a data item and store the signature as separate file in the repository. The other one is to store the signature in the data item's meta data. Both use the hash and sign paradigm for its implementation.

Signature as separate file For this version, a user selects a data item and executes the signing script. The script then hashes the selected data item and asks for a key, that is used to sign the data item. With the given key, preferably provided in a file, the data item is signed. The script generates a file with the signature (for example pkcs7) and stores it in the repository under the name of the data item.

Signature as Meta Data The difference of this concept to the previous one is that instead of saving the signature of the data item in the repository, it is saved in the

[15] Collision resistant means, that there are not two messages who have the same hash.

meta data. Other properties can be saved into the meta data, such as the public key, to use it later for verification. Also information of the signer and his organization can be included.

Concept for signing meta data of a data item

Because of different storing procedures for different meta data back ends, the strategy to sign meta data of an item differs from signing a data item. The concept relies on the and XML signatures (cf.[W3C]).
The work flow of signing meta data is that a user chooses a data item and executes the corresponding script. The script then extracts all meta data of the item and generates a XML file with the meta data in it. The meta data in the XML file should meet standards for meta data serialization, such as Dublin Core(cf. [ISOa]). The XML file is then supposed to be normalized, hashed and signed. The signature is either integrated into the file, meaning added as an extra element to the meta data's XML file, or stored as a separate file.

Implementation of signing data items

As a suitable implementation technique test driven pair programming is chosen. When doing test driven pair programming, one developer designs the test for a feature, while the partner observes. Then the other developer implements the feature, while the test designer observes. The roles are switched constantly, so that both design tests and implement features.
For the implementation different python libraries can be used:

- Cryptopy : library implementing cryptographic algorithms
- keyCzar : toolkit to use cryptographic schemes in their application
- pyxmlsec : general python library used for XML signatures
- pyxmldsig : more convenient interface to pyxmlsec

Due to meeting issues, the feature could only be implemented partly and was not programmed by a pair. So temporarily a file based data item signature is implemented(as described in section 3.3.3). This is used to further test the BeLab service, and is currently sufficient to prove credibility for items.

4 Evaluation of the implementation

This chapter evaluates the integration of the concepts developed in this master thesis. It shows successes and failures. It begins with an evaluation of whether the data management system with the extensions from this thesis meets the defined requirements from chapter 3.1. Next each implementations individually is evaluated based on the usability and adaptability. Also further adjustments are discussed. In the end the software development approaches are evaluated and a recommendation for script development in the DataFinder is extracted.

4.1 DataFinder and laboratory notebook requirements

Table 4.1 evaluates the DataFinder concepts on the requirements from chapter 3.1 and explains how each requirement is integrated into the DataFinder system

Requirement	Implemented?	Details
Chain of events	yes in chapter3.3	provenance for modeling the use case and storing the information
Durability	yes	with extension from chapter 3.3.2, but also former solutions
Immediate documentation	under development	a web portal is implemented
Genuineness	yes customization issue	combination of work flow integration in the DataFinder and the provenance service
Protocol style	yes original	can be added as files to the system
Short notes	yes original	as extra files or meta data to a data item
Verifying results	yes(rudimental)	signing concept and implementation from section 3.3.3
Accessibility	yes original	open source software
Collaboration	yes original	same shared repository for each user, with similar information

Device integration	yes customization issue	integration via script API
Enabling environmental specialisation	yes customization issue	can be customized with scripts and data model
Flexible Infrastructure	yes original	client: platform independent python application server: meta data: WebDAV or SVN (extendable); data: several (extendable)
Individual Sorting	partly under development	customizing the view of the repositories is possible: But saving the settings is in planning
Rights management	yes under construction	the server supports it on the client side, the integration into DataFinder is currently developed
Variety of data formats	yes original	any data format can be integrated, opening them depends on the users system
Searchability	yes original	full text and meta data search
Versioning	yes	SVN as storage backend is developed to enable versioned meta data and data

Table 4.1: Implementation of the laboratory notebook requirements into the DataFinder

The table shows that almost all requirements are either currently met, are integrated in the thesis or currently implemented. This means the DataFinder can be used as laboratory notebook, that supports the concepts of good laboratory practice and therefore the scientific method.
After the implementation the next step is to integrate the system not only as data management system but as laboratory notebook in different organisations. Several institutions are interested and waiting for the implementation for example the Max-Planck-Institutes, DLR institutes and the PTB. [1]
To further improve the laboratory notebook implementation of the DataFinder these features could help:

Mobile version of DataFinder With a mobile version of the data management system, it would ease the scientist's documentation efforts when working on a test site.

[1] Irregularly researchers from these institutes were integrated in the development of the proposed concept.

The scientist could then add notes to newly added data or edit data on-the-fly. The requirement of immediate documentation could be met with this extension.

Automatic generation of reports For many (project) leaders it is interesting to know, what their employees are currently doing, or what the current status of a project is. To check this, they can currently access the data directly. A feature, which summarizes the current reports and gives an intermediate report, could simplify the check up. This feature was implemented in the evaluated laboratory notebook mbllab [mll].

Integrated standard procedures In the GLP a standard procedure defines the workflow for specific machines. In the laboratory notebook mbllab [mll] they are integrated and give the user a guideline for his actions. This could improve the DataFinder laboratory notebook features as well.

More elaborate signing and documenting features Other scientists should discuss results of colleagues. For a more collaborative work situation, the DataFinder needs to be enhanced with more features for interaction of users. So on the one hand a discussion mechanism on data items could be supported, but also some kind of identity card could be left by another scientist, when he signed the data. This could refer to a list of other items he signed or projects he works on. In the evaluated laboratory notebook Notebookmaker [Nbm] a witness principle with library card is integrated. On each notebook page an area is defined, where a scientist can witness (authenticate) an entry. After witnessing the data, the information of the witnessing person is displayed on the corresponding page. The witnessing information is then connected to a library card on which personal information and projects are listed.

A graphical representation Graphical provenance information on the server or in the DataFinder can help to analyse the provenance information. Also the integration of the provenance in the DataFinder helps the user to understand correlations of items.

Configuration options Selecting a provenance system or an Archiving system should be possible. This could be handled with a new option in the data store configuration. Still a dialog to ask for the correct information needs to be implemented.

4.2 Implementation Results

This section gives a short evaluation of the implementation: its usability its adaptability and further adjustments. The definition for each part is:

Usability is defined by the ISO as: "The extent to which a product can be used by specified users to achieve specified goals with effectiveness, efficiency, and satisfaction in a specified context of use." [ISOb] So the analysis of the extension will be on answering the questions: Is the system capable of providing the desired outcome? (Effectiveness)
Is the time and effort of reaching the desired outcome acceptable?(Efficiency)
Is the usage of the script extension to the user's content?(Satisfaction)

Adaptability Subramanian and Chung define adaptability as "the extent to which a software system adapts to change in its environment. An adaptable software system can tolerate changes in its environment without external intervention."[SC99] When analysing the adaptability of the presented script extension, the focus will be on answering: Is the system capable of adapting to a changing environment? What changes need to be done in order for the system to be adapted? [2]

Adjustments is defined in [Weh00]: "a small change made to sth in order to correct or improve it". So in this section the focus is on: Which elements need changing? Which other features are recommended for implementation?

4.2.1 General provenance system "noblivious"

The general provenance system can be used to store information of different provenance models. It operates on a graph database which can be queried with a graph traversal language. The information is send with REST requests. Another REST interface is provided to query the database for information.

Usability The system can store the information successfully. Also querying the service is possible. The time to process the information is acceptable, but depends on the amount of information stored in the database. The user gets success and failure information.

Adaptability The system is capable of supporting different provenance models.

Adjustments One extension of the service could be using SOAP as storing interface. With SOAP more information can be send to the service. The additional information can be used to validate the model or extend the attributes. The additional information could also be semantic information, which then is used to enable improved query results. Another extension to improve the system is the addition of a graphical browser of the database. The graphical browser could help to understand the stored information.

[2] Some of the described adaptations can be done on a central location within the extension, but not guided within the data management system.

4.2.2 Chain of Events: Provenance integration

This feature enables to send provenance information to the provenance system. It is realized with a listener on data import and a dialog, which asks for items influencing the imported data. Then all necessary information is extracted from the data management system and send as REST request to the provenance system, into which the information is stored.

Usability The feature can be used within the DataFinder. It provides the provenance system with the necessary information, which can be used for further processing.
The idea of using a dialog to pop open, after something is being imported helps to ensure, that the user provides the necessary information. Problems could arise, if a user imports several items or a whole folder. This scenario might get unnerving, if too many dialogs pop open and the user has to add too many information. This current design can be seen in figure 4.1.

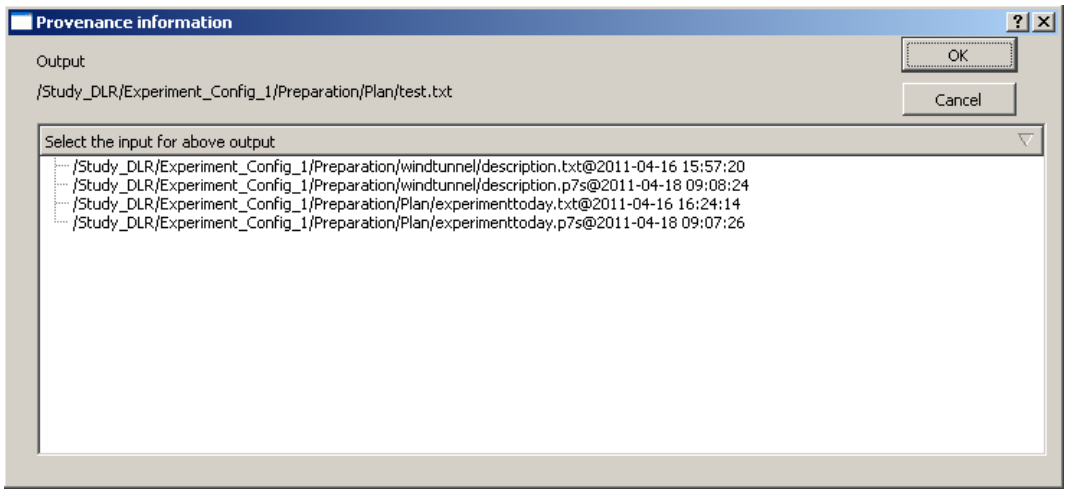


Figure 4.1: Dialog for the chain of events extension

It provides the necessary information, but could be adjusted to better design. One element that needs to be changed for a more satisfying usage, is the box, where the input items are selected. This element should present the input items bundled together according to the location within the system. Otherwise, if a lot of elements exist in the system, an easy orientation in the list is not given.

Adaptability If the provenance implementation is to be used with a new data model in the data management system, it is only partly adaptable. The provenance system with the integrated general interface can be used with a new data model and a different provenance model. The extraction of relevant information trough the script extension is not adaptable, it needs adjustment to the new data model structure.
Adapting the script extension to a new data model means adapting the algorithms that

extract necessary provenance information such as the type of the data item and the process to which the item belongs.

Adjustments Adjustments and further research to improve this feature are:

Pre-selection of input items with semantic evaluation of the service. This could make it easier for the user to chose the correct input. The list could be clustered for relevance, and only files relevant for the provenance model can be chosen. For this the model would need to have a XML representation or something similar, in order to make the decision.[3]

Integration of more actions such as listening on file changes and editing. Also more information that have been added can be included in the provenance service. This would mean the provenance service needs to be adjustable in its attributes that are being set for a specific node.

Automatic extraction of provenance information could be another script implementation, that is used simultaneously and extracts from file headers or meta data the information for the provenance service.

Choosing an actor Currently only a connected user could be extracted. But it could be advantageous to set an actor. With this feature a user could load data into the provenance service from a software or another instrument.

Unregistering making it possible to unregister scripts, even the ones that occur automatically.

A general provenance integration which gives the opportunity to store provenance information detached from import action and data model.

4.2.3 Preservation: Service integration

Preservation includes the extraction of provenance information, packaging the data items into an archive and sending the archive to a service. The service is provided by a project dealing with evidential preservation.
In the end the implementation is split into two different scripts. One script only extracted the items from the provenance service and generates an archive. The other script sends archives to the service. The script is split, because that way it can be used without having a provenance service.

Usability The script extension provides all the feature described above and realizes it effectively.
The communication to the provenance system and the archiving system is rather fast,

[3]The XSD for OPM models is currently drafted: `http://openprovenance.org/model/opmx`(Working draft of the 12th October 2010, seen on the 7th. of March 2011)

and so the user does not have to wait too long.
Starting the process is easily possible, and common to other script extensions. Waiting on the process to finish is sometimes too long. Especially when storing the archive into the data management system, the user is missing a dialog showing the progress. Also new information, such as a new meta data item or data item can only be displayed, if the whole repository connection is being refreshed.

Adaptability The archiving implementation is partly adaptable to a changing environment. One provided implementation makes it possible to only send an archive. There is no need to have a provenance system in order to use the script. The provenance information set needs to be adapted, if there is a provenance system with a different underlying model.
When using a different provenance system, the query for accessing the information needs to be adapted. If the archiving service interface changes, the system needs to be adapted in its generation of the archive, its sending process and its response parsing process.

Adjustments In addition to adjustments, that need to be made because the archiving service is still under development, the following features could be integrated:

- Creation of archives with elements the user provides
- DataFinder interface to provide a provenance query and see a result
- Interface to configure the systems: for each query, the possibility to select a provenance system and an archiving system

4.2.4 Credibilitiy: Signing data

The implemented feature to support authenticity, is a mechanism to issue a detached signature on an item. The feature is the temporary version of a concept, that was presented in chapter 3.3.3.

Usability The signing feature can be used to sign data items. The signature is stored in the system successfully.
Signing a data item is easily accessible for the user and the import is rather fast.
Using the implementation is not completely satisfactory, since it is not possible to chose a signature or signature concept. Also the signature is only visible after a refresh of the whole repository. But the output field in the lower part has a message of the success.

Adaptability It can be used with any data item and does not rely on any data model specific information. Using a different signature is not possible. It can be changed in the script itself.

Adjustments Elements that need to be improved and are part of the developed concept is:

- choice of signature components, such as algorithms, certificates and keys
- verification of a signature within the DataFinder
- integration of signing meta data
- attached signature items (eg. as meta data)

4.3 Script Development Strategies

When developing the script extensions, different implementation approaches were chosen and partly tested.
For the provenance integration, where the requirements was not clearly defined and subject to change, prototyping was chosen. The integration of the BeLab service, which is not a final version of the project, is developed test driven.
The development approaches are now discussed for its impact on the implementation, its advantages and disadvantages.

4.3.1 Prototyping

The prototyping process starts with a basic implementation and with each iteration the prototype either gets more elaborated or implemented totally different. The realisation of such a prototyping process was demonstrated when implementing the provenance feature. The experiences are described here.

Impact on the development When prototyping approach it is possible to see, how the code matured on each iteration step. With each new prototype the code got more elaborate and more modular, because elements were reused and extended.
Due to the changing handling and servicing classes, it was necessary to test more elements in order to maintain functionality. This lead into more test classes with each step.

Advantages Advantage was that problems were tackled in small steps, such as: first try to somehow connect to the service. Then try to get the necessary information from the data management system. Another advantage was that the requirements and problems got clearer and the resulting features were easier to integrate.

Disadvantages Some elements even though they are basic prototypes do not work anymore and need more work to be refactored. So in order to have several versions of one implementation means having to maintain several versions and adapting them. Only relevant or usable versions were adapted until the end.

4.3.2 Test driven development

The feature was implemented test driven. This means before the feature is implemented a test for it was written. The experiences made with this strategy are described here.

Impact on the development It was hard to confine to the strict order of testing then implementing the feature. But in the end it paid off. Because of the multiple test cases it was easy to follow changes and debug the implementation.
Currently only features are implemented, that are truly needed. The whole script is very modular and only has small methods. If another feature is needed, it can be added easily.
Furthermore it was easy to implement another script, which focuses only on part of the implementation, only using parts of the developed classes.

Advantages The code is very well tested and small mistakes can be debugged easily. Also the used services can be tested easily and focus on single features, for example the querying interface of the provenance system.
Later adjustments due to changes in the archiving system can be easily tested and integrated.

Disadvantages Even though tested successfully on the software side, there were still problems when trying to integrate the feature into the data management system. One peculiarity was for example different path delimiters.

4.3.3 Defining a general DataFinder implementation strategy

Each approach was fitting for the selected feature, but a generally good approach can not be recommended. A standard approach to implement a DataFinder feature is hard to define. So each described approach describe an idea of which strategy to use for which scenario in the DataFinder. Also the thesis shows, that different agile development strategies can be used for DataFinder script development.

5 Conclusion

This final chapter summarizes in one section the general outcome of the thesis. In the following section the impact on science and the effects of the thesis on other scientific groups are described.

Presentation of outcome The thesis showed a general approach of "Enabling a data management system to support the good laboratory practice".
Based on prior research from related areas, a requirements analysis for laboratory notebooks was presented. The requirements were extracted from literature sources and a comparison of other approaches of implemented laboratory notebooks. The requirements analysis showed that especially preservation, credibility and a form of traceability is important for scientific work and documentation.
In the implementation part of the thesis three different extensions for a data management system are developed. Each enables the system to meet at least one of the major requirements. The major concepts used are provenance to ensure traceability, a web service that ensures durability and authenticity, as well as signatures for further authenticity. The extensions were developed with different implementation strategies.
The evaluation of the implementation phase, which was described in chapter 4, comes to the end that the extended data management system now supports the required features. It has potential for improvement.
All in all the thesis showed how a data management system can be extended to meet requirements that are extracted from the regulations around the "good laboratory practice".

Impact on other scientists and projects The results of the thesis has different effects on other scientists and research areas.
For the DataFinder community the master thesis provides a general use case for the DataFinder as scientific data management system. It further defines a general data model, which can be used for several research fields. This example use cases introduce new users into the capabilities of the DataFinder.
For the BeLab research team, the integration of their service into an electronic laboratory notebook meant that they are able to proof their concepts within their project. In addition it gives them an example for the usage of their system.
For the research field of provenance, the results of this thesis are most interesting. First of all the thesis identified a new use case of provenance, which embraces different

fields of science. Furthermore this thesis showed an approach of making an application provenance-aware, without touching the core of it. Also the thesis provided a provenance storing system, which can be used independently an underlying model. The presented storage system can be integrated into various architectures.

Bibliography

[BeLa] *BeLab: beweissicheres elektronisches Laborbuch.* http://www.belab-forschung.de

[BeLb] BeLab: *Arbeitspaket 1: Analyse der Forschungspraxis.* internal document,

[Data] *DataFinder - flexible data management.* launchpad.net/datafinder

[Datb] DataFinderTeam: *Langzeitarchivierung im DataFinder.* – https://wiki.sistec.dlr.de/DataFinderProjects/LangzeitArchivierung

[DF08] Davidson, Susan B. ; Freire, Juliana: Provenance and scientific workflows: challenges and opportunities. In: *In Proceedings of ACM SIGMOD*, 2008, S. 1345–1350

[DFG98] DFG: *Proposals for Safeguarding Good Scientific Practice - Recommendations of theCommission on Professional Self Regulation in Science.* Guideline. http://www.dfg.de/foerderung/rechtliche_rahmenbedingungen/gwp/index.html. Version: 1998

[DJCF] Dr. Jean-Claude Franchitti, Computer Science Department Courant Institute of Mathematical S. New York University U. New York University: *Data Mining, Session 4: Laboratory Notebooks.* webpage. http://www.nyu.edu/classes/jcf/g22.3033-002/handouts/LabNotebook.htm

[DLRa] *Deutsches Zentrum für Luft- und Raumfahrt e.V.: About the DLR.* Homepage, . – http://www.dlr.de/en/desktopdefault.aspx/tabid-636/1065_read-1465/

[DLRb] *Deutsches Zentrum für Luft- und Raumfahrt e.V., Simulation and Software Technology: Mission Statement.* Homepage, . – http://www.dlr.de/sc/en/desktopdefault.aspx/tabid-1185/1634_read-3062/

[EBG06] Ebel, H.F. ; Bliefert, C. ; Greulich, W.: *Schreiben und Publizieren in den Naturwissenschaften.* Wiley-VCH, 2006 http://books.google.de/books?id=QbCHsLRuXZMC. – ISBN 9783527308026

[ems] *Electronic Laboratory Notebook.* http://collaboratory.emsl.pnl.gov/software/eln/

[ENo] *E-Notebook*. http://www.cambridgesoft.com/software/details/?ds=9s

[Gal32] Galilei, Galileo: *Dialogue Concerning the Two Chief World Systems*. 1632. – English translation: http://books.google.com/books?id=ST7Y9FFHhrEC&printsec=frontcover&dq=galileo+galilei&hl=en&ei=YZ-fTPH3GoL48AbpvrG2Dg&sa=X&oi=book_result&ct=result&resnum=7&ved=0CE4Q6AEwBg#v=onepage&q&f=false

[GMTM05] Groth, Paul ; Miles, Simon ; Tan, Victor ; Moreau, Luc: *Architecture for Provenance Systems*. http://eprints.ecs.soton.ac.uk/11310/. Version: October 2005

[gre] *Gremlin - G =(V,E)*. – https://github.com/tinkerpop/gremlin/wiki

[HBM+10] Holl, David A. ; Braun, Uri ; Maclean, Diana ; Muniswamy-reddy, Kiran kumar ; Seltzer, Margo I.: Choosing a Data Model and Query Language for Provenance. (2010)

[ISOa] *Information and documentation – The Dublin Core metadata element set*. – http://www.iso.org/iso/iso_catalogue/catalogue_ics/catalogue_detail_ics.htm?csnumber=52142

[ISOb] *ISO 9241 - 11: 1998 Ergonomic requirements for office work with visual display terminals (VDTs) – Part 11: Guidance on usability*. – quote adapted from: http://www.usabilitynet.org

[JK08] Jonathan Katz, Yehuda L.: *Introduction to modern cryptography*. Chapman and Hall/CRC, 2008

[JM07] Jeffries, Ron ; Melnik, Grigori: Guest Editors' Introduction: TDD–The Art of Fearless Programming. In: *IEEE Software* 24 (2007), 24-30. http://doi.ieeecomputersociety.org/10.1109/MS.2007.75. – ISSN 0740–7459

[LAB+05] Ludäscher, Bertram ; Altintas, Ilkay ; Berkley, Chad ; Higgins, Dan ; Jaeger, Efrat ; Jones, Matthew ; Lee, Edward A. ; Tao, Jing ; Zhao, Yang: Scientific Workflow Management and the Kepler System. In: *Concurr. Comput. : Pract. Exper*, 2005, S. 2006

[MCF+09] Moreau, Luc ; Clifford, Ben ; Freire, Juliana ; Gil, Yolanda ; Groth, Paul ; Futrelle, Joe ; Kwasnikowska, Natalia ; Miles, Simon ; Missier, Paolo ; Myers, Jim ; Simmhan, Yogesh ; Stephan, Eric ; Bussche, Jan V.: The Open Provenance Model — Core Specification (v1.1). In: *Future Generation Computer Systems* (2009), December. http://eprints.ecs.soton.ac.uk/18332/

[mll] *mbllab - Das elektronische Laborbuch.* http://www.elektronisches-laborbuch.de/

[MMG+06] Munroe, S. ; Miles, S. ; Groth, P. ; Jiang, S. ; Tan, V. ; Moreau, L. ; Ibbotson, J. ; Vazquez-Salceda, J.: *PrIMe: A Methodology for Developing Provenance-Aware Applications.* http://eprints.ecs.soton.ac.uk/13215/. Version: 2006

[Mor10a] Moreau, Luc: The Foundations for Provenance on the Web. In: *Foundations and Trends in Web Science* 2 (2010), November, Nr. 2–3, 99–241. http://eprints.ecs.soton.ac.uk/21691/

[Mor10b] Moreau, Luc: *OPM Tutorial.* Tutorial at Future Internet Symposium, Berlin, September 2010. – http://openprovenance.org/tutorial/

[MR10] Michael Rubacha, Stephen C. H. Anil K. Rattan R. Anil K. Rattan: A Review of Electronic Laboratory Notebooks Available in the Market Today. In: *Technology Review* (2010)

[MW10] MerriamWebster, Incorporated (Hrsg.): *Merriam-Webster Online Dictionary.* Merriam-Webster, Incorporated, 2010

[Nbm] *NoteBookMaker for PC and Mac , The World Leader in Virtual NoteBooks.* http://www.notebookmaker.com

[neo] *Neo4j - the graph database.* – http://neo4j.org/

[nes] *nestor - detusches Kompetenznetzwerk zur digitalen Langzeitarchivierung.* – http://www.langzeitarchivierung.de/

[OEC97] OECD: *No 1: OECD Principles on Good Laboratory Practice.* http://www.oecd.org/document/63/0,3343,en_2649_34381_2346175_1_1_1_1,00.html. Version: 1997

[ope] *Was ist open enventory?* http://www.chemie.uni-kl.de/forschung/oc/goossen/index.htm?Hauptframe=http://www.chemie.uni-kl.de/goossen/enventory/index_de.html

[OPM] *OPM - Open Provenance Model.* http://openprovenance.org/

[Pas10] Pasche, Tobias: *Parallelen in der Datenrepräsentation zwischen Laborbüchern und sensorerweiterten elektronischen Patientendokumenten*, TU Braunschweig, PTB, Medizinische Hochschule Hannover, Diplomarbeit, 2010

[Pot11] Potthoff, Jan: *Beweissicheres elektronisches Laborbuch (BeLab): Darstellung des BeLab Konzepts.* Workshop - Presentation. http://www.belab-forschung.de/. Version: January 2011

[rce] *RCE - Remote Component Environment.* – http://www.rcenvironment.de/

[SC99] Subramanian, Nary ; Chung, Lawrence: Metrics for Software Adaptability. In: *Applied Technology Division, Anritsu Company* (1999), S. 95–108

[SDS02] Space Data Systems), CCSDS (Consultative C.: Reference Model for an Open Archival Information System (OAIS) / CCDS. 2002. – Forschungsbericht

[Som07] Sommerville, Ian: *Software Engineering.* Pearson Studium, 2007

[SPG05a] Simmhan, Yogesh L. ; Plale, Beth ; Gannon, Dennis: A survey of data provenance in e-science. In: *SIGMOD Rec.* 34 (2005), September, 31–36. `http://doi.acm.org/10.1145/1084805.1084812`. – ISSN 0163–5808

[SPG05b] Simmhan, Yogesh L. ; Plale, Beth ; Gannon, Dennis: A Survey of Data Provenance Techniques. 2005. – Forschungsbericht

[sud] suds: *A lightweight SOAP python client.* – `https://fedorahosted.org/suds/`

[TC09] Tylissanakis, Giorgos ; Cotronis, Yiannis: Data Provenance and Reproducibility in Grid Based Scientific Workflows. In: *Grid and Pervasive Computing Conference, Workshops at the* 0 (2009), 42-49. `http://doi.ieeecomputersociety.org/10.1109/GPC.2009.16`. ISBN 978–0–7695–3677–4

[W3C] W3C: *XML Signature Syntax and Processing (Second Edition).* – `http://www.w3.org/TR/xmldsig-core/`

[Weh00] Wehmeier, Sally (Hrsg.): *Oxford Advanced Learners Dictionary.* 6th. Oxford University Press, 2000

[Wen10] Wendel, Heinrich: *Using Provenance to Trace Software Development Processes*, University of Bonn, Diplomarbeit, 2010

[Wika] Wikipedia: *Edwin Smith Papyrus.* `http://en.wikipedia.org/wiki/Edwin_Smith_papyrus`

[Wikb] Wikipedia: *Open Notebook Science.* `http://en.wikipedia.org/wiki/Open_Notebook_Science`

A Data on CD

The attached CD contains the following items:

- Electronic version of this thesis
- References - as far as possible
- Source Code - as far as it was possible to distribute
- DataFinder information - such as the described model and script extension
- Videos of the execution of the scripts
- Misc:
  - Tables of a comparison from different laboratory notebooks.
  - provenance questions
  - Gremlin queries
  - Master Thesis Proposal
  - Question sheet for interviews
  - Status presentation