Preserving Scientific Processes from Design to Publications

Rudolf Mayer[1], Andreas Rauber[1],
Martin Alexander Neumann[2], John Thomson[3], and Gonçalo Antunes[4]

[1] Secure Business Austria
Favoritenstrasse 16, 1040 Vienna, Austria
{mayer, rauber}@sba-research.at
[2] Karlsruher Institut für Technologie
Kaiserstrasse 12, 76131 Karlsruhe, Germany
mneumann@teco.edu
[3] Caixa Magica Software
Rua Soeiro Pereira Gomes, Lote 1, 8 F, Lisbon, Portugal
john.thomson@caixamagica.pt
[4] INESC ID, Instituto de Engenharia de Sistemas e Computadores, Investigacao e Desenvolvimento, Rua Alves Redol 9, 1000029 Lisbon, Portugal
goncalo.antunes@ist.utl.pt

Abstract. Digital Preservation has so far put its focus mainly on digital objects that are static in their nature, such as text and multimedia documents. However, there is an increasing demand to extend the applications towards dynamic objects and whole processes, such as scientific workflows in the domain of E-Science. This calls for a revision and extension of current concepts, methods and practices. Important questions to address are e.g. what needs to be captured at ingest, how do the digital objects need to be described, which preservation actions are applicable and how can the preserved objects be evaluated. In this paper we present a conceptual model for capturing the required information and show how this can be linked to evaluating the re-invocation of a preserved process.

Keywords: Digital Preservation, Context

1 Introduction

Digital Preservation deals with ensuring the long-term access to digital information objects in the face of changing technologies or designated user communities. So far, the main focus of research in this area has focused on digital objects that are static in their nature, such as text and multimedia documents. There is however the need to extend the research towards dynamic objects (such as interactive art or video games), and beyond to whole processes and workflows. The latter is an emerging topic especially in disciplines such as E-Science, where data-intensive experiments form a core of the research. These experiments and their results need to be verifiable to others in the community. They need to be preserved as researchers need to be able to reproduce and build on top of earlier experiments to verify and expand on the results. A recent report underlines the importance of *Big Data*, noting that it emerges as a new paradigm for scientific discovery that reflects the increasing value of observational, experimental and computer-generated data in virtually all domains, from physics to the humanities and social sciences [1], an aspect which has also been emphasised in the so-called fourth paradigm [3].

Business processes frequently also need to be preserved for issues such as liability cases, where e.g. a company needs to prove that it executed its processes correctly, and faults did not occur because of their manufacturing.

Preserving complete processes and other types of non-static digital objects calls for reassessing and extending current methods and practices. Important research questions to be addressed include

- What needs to be captured at ingest? We need to go beyond single files (and their metadata), up to potentially including and exceeding complete computer systems (or at least a description thereof to be able to recreate them at a later point), and any additional documents that might be needed to understand and operate this process. However, many of todays processes are not limited to single systems but make use of remote services enabled by the Internet of Services (IoS) and Software as a Service (SaaS), and these need to be captured as well .
- How do these digital objects need to be described? We need to characterise not only single files, but several different aspects of the process. This starts from a top-level where organisational parameters need to be described, down to the technical description of the systems the process depends on, including hardware, operating systems, software, and third-party libraries and services. We introduce a *process context model* that addresses these two questions, by proposing a set of aspects that need to be captured and means of storing them in a structured way.
- Which preservation actions are applicable? In practice, there will be a need for combining several different preservation actions, such as migration of specifications and documents, code migration/cross-compilation, or emulation of hardware or software utilised in the process.
- How can a preserved process be verified and evaluated? We need to ensure that the execution of the (modified) process at a later stage is equivalent to the original process. We show the applicability of a recent framework developed for comparing original and emulated versions of digital objects [2].

The remainder of this paper is structured as follows. Section 2 introduces a model to capture contextual information of a process. Section 3 then presents a scientific experiment as a case study for a process, and demonstrates how the context model can be applied. In Section 4, we discuss how verification of preserved processes can be achieved. Finally, we presented conclusions and an outlook on future work in Section 5.

2 Capturing Process Context

Todays digital preservation approaches focus on preserving (mostly static) digital objects, and additional information about these objects. In terms of the Open Archival Information System (OAIS) [4], this additional information can be denoted as *Representation Information*, i.e. the information needed so that Designated Communities can understand the digital object at a later point in time, as well as Preservation Description Information (PDI), i.e. is the additional metadata needed to manage the preservation of the objects.

The context of information needed for preserving processes is considerably more complex, as it not only requires dealing with the structural properties of information, but also with the dynamic behaviour of processes.

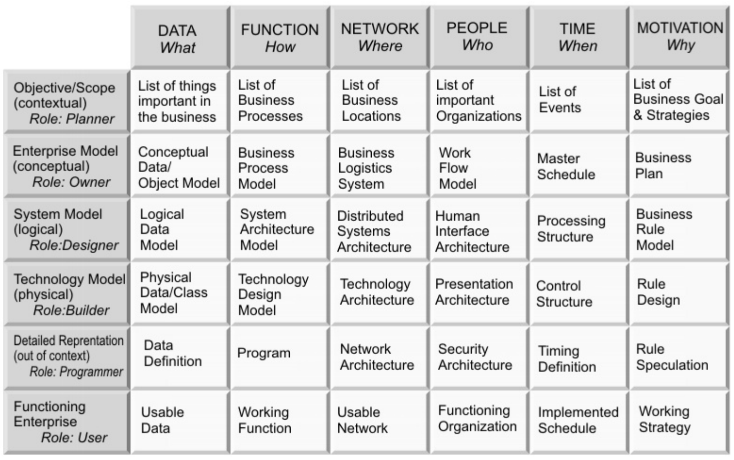
A successful digital preservation of a business process requires capturing sufficient detail of the process, as well as its context, to be able to re-run and verify the original behaviour at a later stage, under changed and evolved conditions. This may include different stakeholders and parties, different or evolved enabling technologies, different system components on both hardware and software levels, changed services (or terms of service) by external service providers (potentially caused as well by a change in technologies or components), and differences in other aspects of the context of the business process.

To enable digital preservation of business processes, it is therefore required to preserve the set of activities, processes and tools, which all together ensure continued access to the services and software which are necessary to reproduce the context within which information can be accessed, properly rendered and validated.

To address these challenges, we have devised a context model to systematically capture aspects of a process that are essential for its preservation and verification upon later re-execution. The model consists of approximately 240 elements, structured in around 25 major groups. The model is implemented in the form of an ontology, which on the one hand allows for the hierarchical categorisation of aspects, and on the other hand shall enable reasoning, e.g. over the possibility of certain preservation actions for a specific process instance. The ontology is authored in the Web Ontology Language (OWL). We developed a set of plug-ins for the Protégé ontology editor to support easier working with the model.

Basically, this context model corresponds to some degree to the representation information network [5], modelling the relationships between an information object and its related objects, be it documentation of the object, constituent parts and other information required to interpret required to interpret the object. Here this is extended to understand the entire context within which a process, potentially including human actors, is executed, forming a graph of all constituent elements and, recursively, their representation information.

The model was derived from the combination of two approaches. The first approach was a top-down approach, utilising models from enterprise architecture frameworks. Most prominently, we employed the Zachman Framework [11], a schema for classifying artefacts. It consists of a two-dimensional classification



	DATA *What*	FUNCTION *How*	NETWORK *Where*	PEOPLE *Who*	TIME *When*	MOTIVATION *Why*
Objective/Scope (contextual) *Role: Planner*	List of things important in the business	List of Business Processes	List of Business Locations	List of important Organizations	List of Events	List of Business Goal & Strategies
Enterprise Model (conceptual) *Role: Owner*	Conceptual Data/ Object Model	Business Process Model	Business Logistics System	Work Flow Model	Master Schedule	Business Plan
System Model (logical) *Role:Designer*	Logical Data Model	System Architecture Model	Distributed Systems Architecture	Human Interface Architecture	Processing Structure	Business Rule Model
Technology Model (physical) *Role:Builder*	Physical Data/Class Model	Technology Design Model	Technology Architecture	Presentation Architecture	Control Structure	Rule Design
Detailed Reprentation (out of context) *Role: Programmer*	Data Definition	Program	Network Architecture	Security Architecture	Timing Definition	Rule Speculation
Functioning Enterprise *Role: User*	Usable Data	Working Function	Usable Network	Functioning Organization	Implemented Schedule	Working Strategy

Fig. 1: Zachman Enterprise Architecture Framework

matrix, where the columns represent communication questions (why, how, what, who, where, and when), and the rows transformations, as shown in Figure 1.

For a bottom-up approach, existing taxonomies such as the PREMIS data dictionary [9], as well as a scenario based analysis, have been employed. A number of scenarios for business process preservation, from various domains, have been devised, and their relevant aspects been identified. One of these processes, a scientific experiment, will be described as case study in Section 3.

Two sections of this model are depicted in Figure 2. Each item represents a class of aspects, for which a specific instance of the context model then creates concrete members, which are then related to each other with properties.

Figure 2(a) details aspects on software and specifications. Technical dependencies on software and operating systems can be captured and described via CUDF (Common Upgradeability Description Format) [10] for systems which are based on packages, i.e. where there is a package universe (repositories) and a package manager application. Such an approach allows to capture the complete software setup of a specific configuration, which then can be recreated. Related to the software installed, capturing information on the licences associated to them allows for verifying which preservation actions are permissible for a specific scenario. Software and/or its requirements are formally described in specification documents. Specific documents for a process are created as instances of the appropriate class, and related to the software components they describe.

Configuration, also depicted in Figure 2(a), is another important aspect, closely related to software (and hardware). Maybe even more than the specific version of a software utilised might influence the process outcome, can the specific configuration applied alter the behaviour of an operating system or software component. Capturing this configuration might not always be easy. Again, in

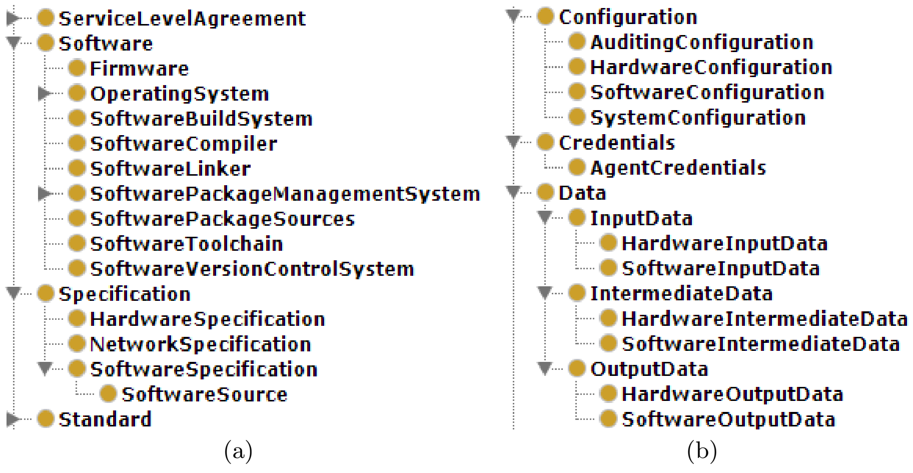


Fig. 2: Sections of the Context Model

systems that rely on packages for their software, these packages tend to provide information about default locations for configuration files, which might be a start for capturing tools.

Another important aspect of the context model deals with several types of data consumed and created by a process, as seen in a section of Figure 2(b). We distinguish between data that originates from hardware or software, and whether this data is input to or output of the process, or created and consumed inside the process, i.e. output from one process step and input for another. Capturing this data is an important aspect in verifying that a re-execution of a process yields the same results as the original process, as we will detail in Section 3. It may be easily captured if the process is formally defined in a workflow engine. In other cases, it may be more difficult to obtain, e.g. by observing network traffic or system library calls.

Other aspects of the model cover for example human resources (including e.g. required qualifications for a certain role), actors, or legal aspects such as data protection laws. Location and time-based aspects need to be captured for processes where synchronisation between activities is important. Further important aspects are documentation and specifications, on all different levels, from high-level design documents of the process, use-case specifications, down to test documents, etc.

While the model is very extensive, it should be noted that a number of aspects can be filled automatically – especially if institutions have well-defined and documented processes. Also, not all sections of the model are equally important for each type of process. Therefore, not every aspect has to be described in most detail.

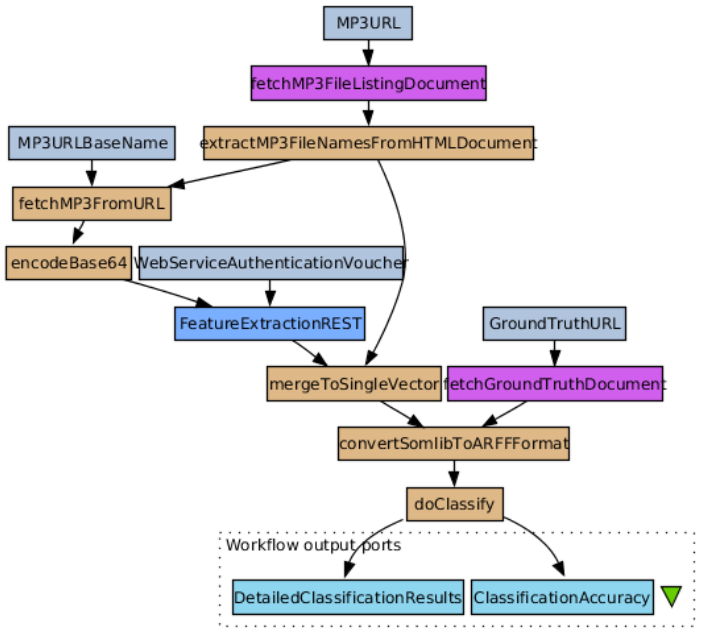


Fig. 3: Scientific workflow modelled in the Taverna Workflow engine

3 Case Study: Scientific Experiment

The process used in our case study is a scientific experiment in the domain of data mining, where the researcher performs an automatic classification of music into a set of predefined categories. The experiment involves several steps, roughly in this order:

– Music data is acquired from online content providers.
– For this music data, genre assignments are obtained from websites such as Musicbrainz.org.
– A web-service is employed to extract numerical features describing certain characteristics of the audio files.
– The numerical description and the genre assignments are combined.
– This forms the basis for learning a machine learning model, which is finally employed to predict genre labels for unknown music.

Besides these steps, several scripts are used to convert data formats and for other similar tasks.

An implementation of this scientific experiment workflow with Taverna[7] is given in Figure 3. Taverna is one of the most prominent scientific workflow management systems (SWMS) existing today. It allows scientists to easily combine services (remote services or programs/scripts) and infrastructure for their research, and have a complete and documented model of their experiment process.

The process depends on several components that are not under direct control of the researcher, most notably the web service used to extract the numeric feature representation from the audio files. Also the software used for machine learning frequently has new releases, and changes in the process outcome might be due to bugs being solved or introduced. Thus there is the risk that that the workflow might lead to different results at a later execution time. In such an event, it is relevant to know that (a) something has changed, and (b) in which component of the workflow it has changed.

In the Taverna implementation, scripts and commands that would have been executed with the shell of the operating system have been migrated to scripts in the Taverna-supported language *beanshell*, based on the Java programming language. Note that this already constitutes some form of migration of the original process, at the same time rendering it more platform independent. Taverna is capable of capturing the data exchanged between the process steps as provenance data, which can be stored in the Taverna-specific format Janus [6] (the also available Open Provenance Model format [8] contains only information of the invoked process steps, but not the actual data).

Through a series of iterations, we modelled this scientific experiment in the above presented Context Model. Figure 4 gives an overview on the concrete instances and their relations identified as relevant aspects of the business process context.

As the scientific experiment is a process mostly focusing on data processing, the majority of the identified aspects are in the technical domain – software components, external systems such as the web service to extract the numerical audio features from, or data exchanged and their format and specification. However, also goals and motivations are important aspects, as they might heavily influence the process. As such, the motivation for the providers of the external systems is relevant, as it might determine the future availability of these services. Commercial systems might be more likely to sustain than services operated by a single person for free.

Another important aspect in this process are licences – depending on which licence terms the components of our process are released under, different options of preservation actions might be available or not. For closed-source, proprietary software, migration to a new execution platform might be prohibited.

A central aspect in the scientific process is the AudioFeatureExtractionService, i.e. the remote web-service that provides the numeric representation for audio files. The service needs as input files encoded in the MP3 format (specified by the ISO standard 11172-3). More specifically, as they are binary files, they need to be further encoded with Base64, to allow for a data exchange over the HTTP protocol. The web-service further accepts a number of parameters that control the exact information captured in the numeric representation; they are specified in the AudioFeatureExtractionSpecification, which for example also covers a detailed information on how the extraction works. The service requires an authorisation key. The operator of the web-service provides the service for free, but grants authorisation keys that are non-transferable between different

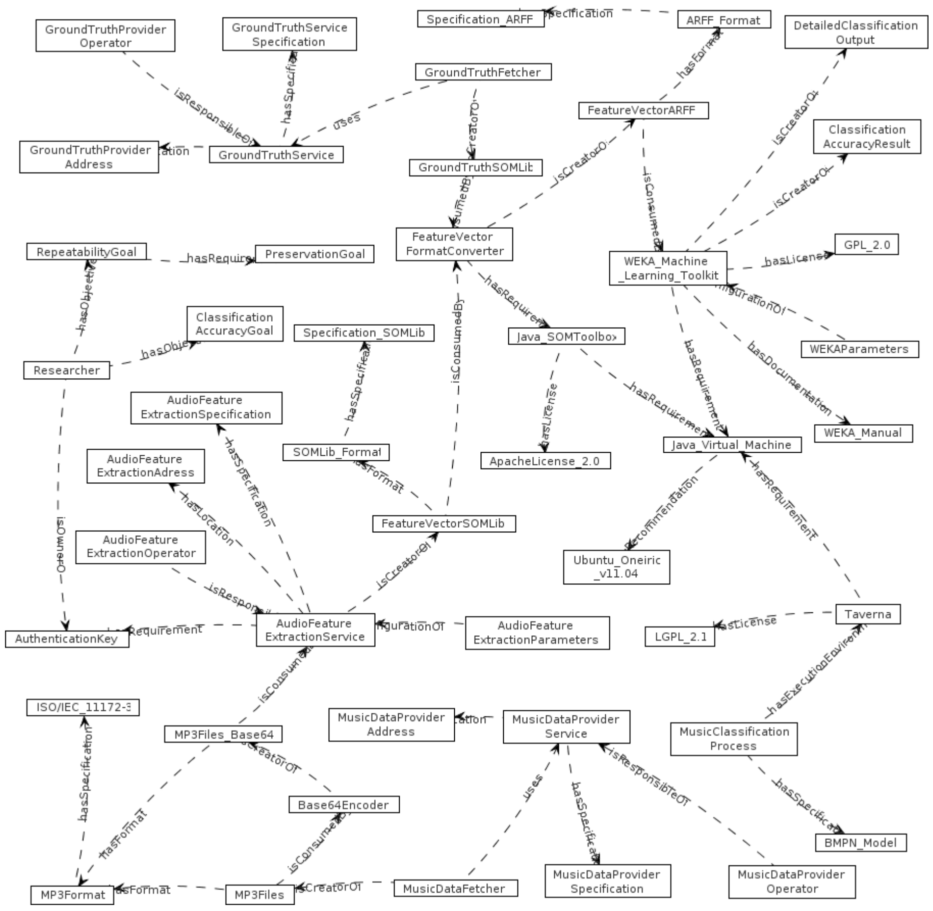


Fig. 4: Context of scientific workflow captured in the Context Model

researchers. Finally, the feature extraction service provides the numeric description as ASCII file, following the SOMLib format specification.

As a software component used locally, the WEKA machine learning toolkit requires a Java Virtual Machine (JVM) platform to execute. The JVM in turn is available for many operating systems, but has been specifically tested on a Linux distribution, Ubuntu, version “Oneiric” 11.04. WEKA requires as input a feature vector in the ARFF Format, and a set of parameters controlling the learning algorithm. These parameters are specified in the WEKA documentation. As output result, the numeric performance metric “accuracy” is provided, as well as a textual, detailed description of the result. WEKA is distributed under the terms of the open-source GNU Public License (GPL) 2.0, which allows for source code modifications.

After this experimentation process, a subsequent process of result analysis and distillation is normally performed, taking input from the experiment outcomes, and finally leading to a publication of the research in the form of e.g. a conference or journal paper. This, again, may be modelled either as a single information object (the paper) connected to the process, and thus to all data and processing steps that led to the results published, or as a more complex process in its own, specifically if a paper reports on meta-studies across several experiment runs. This can be modelled via connections between the individual components, e.g. the paper (and its supporting representation information network and entire context model) and the process context model.

4 Evaluation of Preserved Processes

One important aspect in Digital Preservation is the verification that the preserved object is equivalent to the original one, i.e. that the identified significant properties are equal. This is also valid when preserving business process – finally, after capturing the context of a process, one needs to asses whether two executions of the specific business process (or workflow), are equivalent.

A framework for evaluating whether two versions of a digital object are equivalent is presented in [2]. To this end, the authors propose to measure and compare significant properties of the original and modified object. Important steps in the this framework include a (1) description of the original environment, (2) the identification of external events influencing the object's behaviour, (3) the decision on what level to compare the two objects, (4) recreating the environment, (5) applying standardised input to both environments, and finally (6) extracting and (7) comparing the significant properties. Even though [2] focuses mostly on emulation of environments, the principles have also been discussed and are applicable specifically for entire processes, and will work virtually unchanged also for migration approaches, when complex objects are transformed e.g into a new file format version.

Relating to the framework for comparing systems presented above, external events (2) in the scientific experiment can be the external system used – the ones for providing the data, and the ones providing functionality such as the feature extraction web-service. The level to compare two instances of a process (3) is primarily on the interface between the different process steps. Provenance data captured during process execution can be utilised to apply the standardised input (5) to the current instance of the workflow. Significant properties in the process are mostly the data output from the overall process, and from each intermediate step. Provenance data should be able to capture these significant properties (6).

The recorded provenance data can be utilised to verify whether the new version of the process still renders the same results. To this end, as the evaluation framework suggests, one can automatically reapply the inputs and verify the recorded outputs, similar to what would be performed in automated software testing. An example of the provenance data recorded for the two process outputs, the percentage of correctly classified instances, and the detailed classification results, are given in Listings 1.1 and 1.2. Note that some unique identifiers, such as URLs as namespaces, and identifiers for the workflow and specific data elements, have been abbreviated for space reasons.

```
<rdf:Description rdf:about="{nsTaverna}/2010/workflow/{idWF}/processor/
      MusicClassificationExperiment/out/ClassificationAccuracy">
  <janus:has_value_binding rdf:resource="{nsTaverna}/2011/
        data/{idDataGrp}/ref/{idDataPort0}"/>
  <rdfs:comment rdf:datatype="{nsW3}/2001/XMLSchema#string">
    ClassificationAccuracy
  </rdfs:comment>
  <janus:is_processor_input rdf:datatype="{nsW3}/2001/XMLSchema#boolean">
    false
  </janus:is_processor_input>
  <janus:has_port_order rdf:datatype="{nsW3}/2001/XMLSchema#long">
    0
  </janus:has_port_order>
  <rdf:type rdf:resource="http://purl.org/net/taverna/janus#port"/>
</rdf:Description>

<rdf:Description rdf:about="{nsTaverna}/2011/data/{idDataGrp}/ref/{idDataPort0}">
  <rdfs:comment rdf:datatype="{nsW3}/2001/XMLSchema#string">
    80.0
  </rdfs:comment>
  <janus:has_port_value_order rdf:datatype="{nsW3}/2001/XMLSchema#long">
    1
  </janus:has_port_value_order>
  <janus:has_iteration rdf:datatype="{nsW3}/2001/XMLSchema#string">
    []
  </janus:has_iteration>
  <rdf:type rdf:resource="http://purl.org/net/taverna/janus#port_value"/>
</rdf:Description>
```

Listing 1.1: Example provenance data of Taverna for the process output *ClassificationAccuracy* (cf. Figure 3). The first RDF Description element defines the output port *ClassificationAccuracy*, the second element contains the actual value of "80.0". Note that some identifiers have been abbreviated, marked by {...}

```
<rdf:Description rdf:about="{nsTaverna}/2010/workflow/{idWF}/processor/
      MusicClassificationExperiment/out/DetailedClassificationResults">
  <janus:has_value_binding rdf:resource="{nsTaverna}/2011/
        data/{idDataGrp}/ref/{idDataPort1}"/>
  <rdfs:comment rdf:datatype="{nsW3}/2001/XMLSchema#string">
    DetailedClassificationResults
  </rdfs:comment>
  <janus:is_processor_input rdf:datatype="{nsW3}/2001/XMLSchema#boolean">
    false
  </janus:is_processor_input>
  <janus:has_port_order rdf:datatype="{nsW3}/2001/XMLSchema#long">
    0
  </janus:has_port_order>
  <rdf:type rdf:resource="http://purl.org/net/taverna/janus#port"/>
</rdf:Description>

<rdf:Description rdf:about="{nsTaverna}/2011/data/{idDataGrp}/ref/{idDataPort1}">
  <rdfs:comment rdf:datatype="{nsW3}/2001/XMLSchema#string">
      1   2:Hip-Hop   2:Hip-Hop        0.667 (3.359461)
      2   2:Hip-Hop   2:Hip-Hop        0.667 (3.294687)
      3  1:Classica  1:Classica        0.667 (2.032687)
      4      3:Jazz      3:Jazz        0.667 (2.536849)
      5  1:Classica  1:Classica        0.667 (1.31727)
      6  1:Classica      3:Jazz   +    0.667 (3.46771)
      7      3:Jazz  1:Classica   +    0.333 (2.159764)
      8   2:Hip-Hop   2:Hip-Hop        0.667 (3.127645)
      9      3:Jazz      3:Jazz        0.667 (3.010563)
     10   2:Hip-Hop   2:Hip-Hop        0.667 (4.631316)
</rdfs:comment>
```

Listing 1.2: Example provenance data of Taverna for the process output *DetailedClassificationResults* (cf. Figure 3). The first RDF Description element defines the output port *DetailedClassificationResults*, the second element contains the actual value, one entry for each file tested, with the actual class, the predicted class, and the confidence of the classifier in the prediction. Note that some identifiers have been abbreviated, marked by {...}

Each listing contains two RDF *Description* elements, where the first one defines the output port, and contains as a sub-element the identifier of the element containing the actual value, which is the second *Description* element in both listings. With the identifiers used in the *rdf:about* attributes, it is possible to uniquely identify the process step (and iteration, if the step is looped over) the data originates from.

The provenance data can further be used for implementing a watch service for software and external service dependencies, e.g. by periodically executing the process with all historic recordings of previous executions, either as a complete process, or for each process step individually.

5 Conclusions and Future Work

Preservation of business processes is a challenge that has so far not been tackled by Digital Preservation research. However, there is a need to re-run processes at a later time, for example in areas such as E-Science, or for liability issues when one needs to show that a process was executed correct.

In this paper, we therefore presented a model to capture important aspects of a process and its context, enabling to preserve the process for later re-execution. We showed how this model can be applied to describe a process from the E-Science domain, a scientific experiment in the area of machine learning.

Many aspects of the context model can be populated automatically, such as the software dependencies and their licences. The evaluation and verification of the preserved process is a challenge, but it can be easier performed in fully documented processes. This holds especially true if these processes are modelled and executed in a workflow engines, where one can automatically record provenance data to verify whether later executions of the process render the same results.

Future work includes development of software components to extract more aspects of the business process in an automatic way, and modules to verify the preserved process.

A research challenge will be the complexity of preserving processes which might be composed of hundreds of parts and components, which all might have dependencies on each other. This will require more refined approaches than traditional Digital Preservation offers today.

6 Acknowledgments

Part of this work was supported by the project TIMBUS, partially funded by the EU under the FP7 contract 269940.

References

1. K. Glinos. E-infrastructures for big data: Opportunities and challenges. *ERCIM News*, 2012(89), 2012.

2. M. Guttenbrunner and A. Rauber. A Measurement Framework for Evaluating Emulators for Digital Preservation. *ACM Transactions on Information Systems (TOIS)*, 30(2), 2012.
3. T. Hey, S. Tansley, and K. Tolle, editors. *The Fourth Paradigm: Data-Intensive Scientific Discovery*. Microsoft Research, Redmond, Washington, 2009.
4. International Organization For Standardization. OAIS: Open Archival Information System - Reference Model. 2003. Ref. No ISO 14721:2003.
5. Y. Marketakis and Y. Tzitzikas. Dependency management for digital preservation using semantic web technologies. *International Journal on Digital Libraries*, 10:159–177, 2009.
6. P. Missier, S. S. Sahoo, J. Zhao, C. A. Goble, and A. P. Sheth. *Janus*: From workflows to semantic provenance and linked open data. In *Proceedings of the International Provenance and Annotation Workshop (IPAW2010)*, pages 129–141, Troy, New York, USA, June 15–16 2010.
7. P. Missier, S. Soiland-Reyes, S. Owen, W. Tan, A. Nenadic, I. Dunlop, A. Williams, T. Oinn, and C. Goble. Taverna, reloaded. In *Proceedings of the 22nd international conference on Scientific and Statistical Database Management*, SSDBM'10, pages 471–481, Berlin, Heidelberg, June 2010. Springer-Verlag.
8. L. Moreau, J. Freire, J. Futrelle, R. E. Mcgrath, J. Myers, and P. Paulson. *Provenance and Annotation of Data and Processes*, chapter The Open Provenance Model: An Overview, pages 323–326. Springer-Verlag, Berlin, Heidelberg, 2008.
9. PREMIS Editorial Committee. Premis data dictionary for preservation metadata. Technical report, March 2008.
10. R. Treinen and S. Zacchiroli. Description of the CUDF Format. Technical report, 2008. http://arxiv.org/abs/0811.3621.
11. J. A. Zachman. A framework for information systems architecture. *IBM Systems Journal*, 26(3):276–292, 1987.