The PERICLES Process Compiler: Linking BPMN Processes into Complex Workflows for Model-Driven Preservation in Evolving Ecosystems

Noa Campos-López[1,2] and Oliver Wannenwetsch[2]
[1]*University of Göttingen, Göttingen, Germany*
[2]*Gesellschaft für Wissenschaftliche Datenverarbeitung Göttingen mbH (GWDG), Göttingen, Germany*

Keywords: Digital Preservation, Model-Driven Preservation, Process Model, Digital Ecosystem, Digital Art, BPMN, RDF.

Abstract: Understanding and reusing archived digital information after a decade of technological advancements, constant developments and changes to software and hardware is a very complex and time consuming task. Without a clear documentation and records of changes, research on the creation and modification of information systems and its associated data is almost impossible. To ease these problems, the EU-funded project PERICLES follows a model-driven preservation approach, where the digital ecosystem is modelled and updated constantly to handle changes for generating a holistic record on information systems involvement. For realising this approach, we present in this paper the PERICLES Process Compiler, which consumes descriptive models of changing environments and evolving semantics to generate executable workflows that are capable of re-enacting changes in information systems and mitigate their impact. By this, our contribution narrows the gap between the theory of model-driven preservation and its application in real information system environments.

1 INTRODUCTION

Digital content and its associated metadata are generated and used across different phases of the information lifecycle and in a continually evolving environment. Therefore, the concept of a fixed and stable *final* version that needs to be preserved becomes less appropriate. In order to deal with technological change and obsolescence, long-term sustainability requires to address changes in context, as well as changes in semantics. These changes involve *semantic drift* that arises from changes in language and meaning or disciplinary and societal changes that affect the practices, attitudes and interests of users and stakeholders (Davenport and Cronin, 2000).

Capturing and maintaining this information throughout the data lifecycle, together with the complex relationships between the components of the digital ecosystem as a whole, is the key to an approach based on *preservation by design* (Kowalczyk, 2008). For this, models that capture intents and interpretive contexts associated with digital content are subject of current research on preservation, which enables content to remain relevant to new communities of users.

The EU-funded project PERICLES (Promoting and Enhancing Reuse of Information throughout the Content Lifecycle taking account of Evolving Semantics) follows the approach of modelling, capturing and maintaining details of digital ecosystems for complex information systems on digital artefacts (PERICLES, 2015). For this, PERICLES follows the principle of *model-driven preservation (MDP)* that includes a holistic capturing, analysing, and updating of software environments with all applied changes,

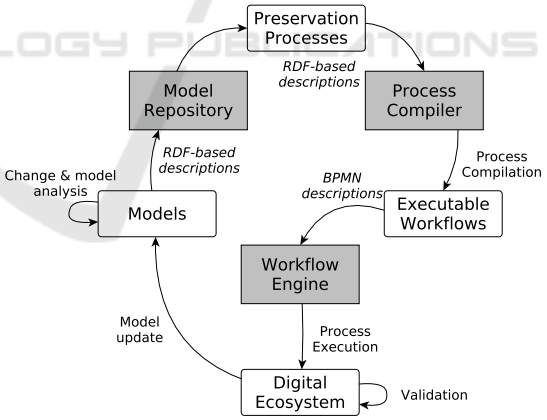

Figure 1: PERICLES Model-Driven Preservation with automatic process compilation.

Campos-López, N. and Wannenwetsch, O.
The PERICLES Process Compiler: Linking BPMN Processes into Complex Workflows for Model-Driven Preservation in Evolving Ecosystems.
In *Proceedings of the 12th International Conference on Web Information Systems and Technologies (WEBIST 2016) - Volume 1*, pages 76-83
ISBN: 978-989-758-186-1

involved processes and interactions with digital data over evolving semantics (Vion-Dury, 2015). This complete capturing is important, because *change* does not only lead to a loss of functionality in software systems but also to a loss of meaning, understanding and reusing of digital information (Vion-Dury et al., 2015).

In this paper, we give an overview on the *PERICLES Process Compiler*, a core software component that helps to realise and partially automate the MDP approach by compiling linked RDF-based processes into executable BPMN workflows. By this, we deliver contributions on preservation action management by linking the domain of MDP to the executable workflow domain.

The paper is structured in eight sections. First, we provide an insight into existing preservation techniques by process composition, and environment and semantic modelling in Section 2. Then, we introduce the MDP approach in PERICLES in Section 3, and describe our contribution on linking RDF-based processes by extending the ecosystem model with three new concepts in Section 4. Latter, we illustrate this approach with a case study on preserving digital art in Section 5 and present the technical realisation of the Process Compiler in Section 6. We conclude our paper and provide an insight into future features of the Process Compiler in Sections 7 and 8, respectively.

2 RELATED WORK

In literature, different aspects of preservation and curation of software environments have been discussed in recent years. While the focus of recent literature is on technical challenges of preservation or on high level approaches, publications which link both domains together that are usable for non-software experts are still underrepresented. For preserving software environments, different approaches have been introduced, such as performance model-based preservation (Matthews et al., 2010). However, this approach coming from the area of libraries does not reflect the changing nature of software environments that has been recognized by the software development community (Barateiro et al., 2012). As software is just one essential part in complex scenarios, preservation processes aim on higher semantics of preserving data, software environments and evolving semantics (Neumann et al., 2011).

New approaches that link business process models with metamodels and ontologies have appeared in recent years. These approaches cover semantic interoperability of process models by adding higher precision in modelling and enable semantic querying of large process model repositories (Höfferer, 2007) and (Smith et al., 2012).

Other approaches cover the employment of semantic process models by linking model elements with concepts from formal ontologies. Linking those domains together improves the representation of process models, enables process validation at semantic level, and simplifies the process execution (Thomas and Fellmann, 2007). As processes are also subject to changes in evolving environments during their lifecycle, approaches that manage these changes and allow the reconfiguration of process models are of great interest. Gottschalk et al. propose configurable workflow models by extending workflow modelling languages with configuration elements and their configurations (Gottschalk et al., 2008), whereas Koliadis et al. combine high-level conceptual models with business process models to populate changes in both context (Koliadis et al., 2006). However, these approaches do not consider changes in semantics and this is the gap, where we provide a contribution with MDP and process compiling. Many publications on process composition based on compiling ontological descriptions into executable process models can be found in literature. These approaches allow process reuse, and facilitates the process modelling, reasoning and matching. However, they are mostly focused on specific domains, i.e. Web Service composition (Rao et al., 2004), (Rao and Su, 2005), (Martin et al., 2005), and (Sirin et al., 2002), or scientific workflow compilation (Ludäscher et al., 2003).

From the side of modelling computation environments for scientific applications, cloud computing added new impulses of deploying infrastructure models into executable cloud environments. One essential standard in this setting is Topology and Orchestration Specification for Cloud Applications (TOSCA) from the OASIS, which is used to describe topologies of cloud based web services, their relations, components, and processes used to manage them (Binz et al., 2012). Similar to PERICLES, TOSCA is a current research topic in the area of executable environments for scientific applications in the cloud (Glaser, 2015).

3 MODEL-DRIVEN PRESERVATION

In this line-up, the EU-funded project PERICLES is looking at the whole picture from the different angle of modelling and builds upon the results on performing digital preservation. The research focus of PERICLES is on understanding the implication

of changing ecosystems in contrast to specific challenges on system changes over time, as it was formulated five years ago (Brocks et al., 2010).

The model-driven preservation approach employed and extended in PERICLES consists of a set of interrelated ontologies and models described as Resource Description Framework (RDF) *triples*. One of these models is the **ecosystem model**, describing the digital ecosystem in which digital resources are produced, stored, modified and processed. The PERICLES project, as context where our work is conducted, defines as digital ecosystem a set of interrelated entities that evolve on time (PERICLES, 2015). This set comprises digital objects, user communities, policies, processes, technical systems, and the relations and interactions between them. If changes in the ecosystem are applied, i.e. an update of the operating system, an update of the ecosystem model is required. For managing the evolution and change of the digital ecosystem, the ecosystem model is fundamentally built on the **Linked Resource Model (LRM)**, which provides a framework to build the dependencies between entities as a set of evolving linked resources (Vion-Dury et al., 2015). The LRM allows to define the semantics of a change in terms of its intention and manage its impact on the ecosystem, key to long-term preservation, understanding and reusing of digital information. While the LRM and ecosystem model provide a domain-independent basis to model digital resources in changing environments and evolving semantics, **domain-specific ontologies** are used to describe specific ecosystems for the different use cases.

Regular checks and analyses of models together with a validation of the digital ecosystem against its representation in the ecosystem model are necessary to detect changes in data objects, technology, context, semantics and user communities. When a change occurs, *preservation actions* are applied on the ecosystem to preserve the digital resources and mitigate the impact of change (c.f. Figure 1). Hence, the set of models and the derived preservation actions allow to trace all changes in digital systems and information processing and allow an (semi-) automatic reconstruction of digital ecosystems, capable of executing ancient software in their original environment. Preservation actions operate over the digital ecosystem, and change and evolve as it does. Therefore, we model these preservation actions as a set of process entities interrelated with other entities in the digital ecosystem by extending the ecosystem model with new concepts, namely aggregated process, atomic process and implementation entities (c.f. Subsec. 4.1 and 4.2). A *process entity* in the ecosystem model is a high-level description of a process: which function it performs, which data it consumes and produces, and which services it runs, while delegating the low-level details to be described in an associated file using a well suited notation (e.g. Business Process Model and Notation (BPMN)). This BPMN file can be then executed over real information systems to perform the preservation actions. This approach allows to use the same technology to reason, manage, query and store all models and entities used in the preservation system, including the LRM functionalities of change management and semantic versioning. Thus, our preservation system takes advantage of the benefits provided by semantic process models (c.f. Sec. 2) without having to develop a complete new process ontology.

4 THEORETICAL APPROACH

The PERICLES Process Compiler that we describe in this paper allows to transform and combine RDF-based descriptions of preservation processes into executable BPMN workflows (c.f. Fig. 1). This can be done because we have included three new concepts in the ecosystem model: *aggregated processes*, *atomic processes* and *implementations*. These entities can be used to easily design new preservation actions by process combination, while enabling process validation within the ecosystem model at semantic level and within the process model specification (e.g. BPMN) at implementation level.

4.1 Aggregated and Atomic Process Entities

An *aggregated process* is a process that can be described as a combination of other process entities. For this, both process flow and data flow are defined. They describe how processes, and produced and consumed data are connected within each other.

In contrast, an *atomic process* is a process that cannot be decomposed in other process entities, even if its execution may include more than one task. Atomic processes have information about the operators, namely the entities responsible of performing the tasks involved in the execution of the workflow.

This differentiation in aggregated and atomic processes provides more flexibility to create new preservation actions while enabling the reusability of reliable ecosystem processes. It facilitates a clearer visualization of dependencies between entities in the ecosystem and allows us to keep track of which processes are used by others. It also reduces the scope of change. For example, if the technology of a service

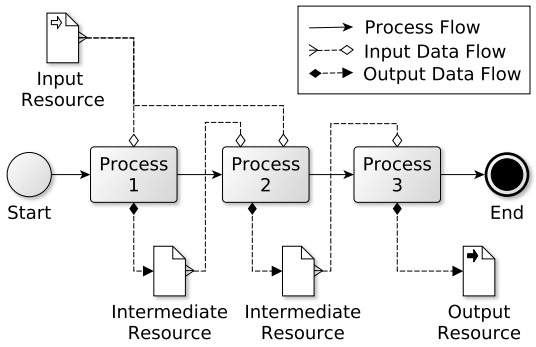

Figure 2: Process flow and data flow in an aggregated process composed of three processes.

used by an atomic process changes, only the implementation files need to be recompiled, while aggregated process descriptions and dependencies remain valid.

As being part of the digital ecosystem, process entities and their dependencies are described and modelled in the ecosystem model (c.f. Sec. 3). The ecosystem model assigns *attributes* to each entity type, describing its instances with identifiers, human readable names, versions and other necessary attributes. For modelling the behaviour and relationships of a process in the digital ecosystem, the ecosystem model provides a set of dependencies - which inherits from the LRM dependency definition - between processes and other entities, i.e. *isInputOf*, *isOutputOf* or *isImplementationOf*. In the following subsections, a brief overview of the process type attributes is given, to explain the input side of the PERICLES Process Compiler.

4.1.1 Common Process Attributes

Originated from the same abstract process entity, both process types of aggregated and atomic processes share a common set of attributes. In the following, we explain the most important attributes:
Identifiers are provided for machinery identification with a unique identifier and a human-readable name.
Descriptions are provided for explaining the process, its purpose, which operation/functionality it performs, over which entities and under which conditions.
Inputs are describing the inputs that can be inserted into a process. They are provided as a list of input slots $*(0..n)$, specifying for processes a set of variables with a unique ID, a human-readable name, an input data type, and a flag marking the input as required or optional.
Outputs are the entities generated by a process. Similar to inputs, they are organized as a list of slots $*(0..n)$, specifying for processes a set of variables with a unique ID, a human-readable name, and an output data type.
Implementations are containing the entities with the low-level description of the process (c.f. Subsec. 4.2.).

4.1.2 Atomic Process Attributes

Atomic processes as the smallest unit in the decomposition for the Process Compiler consist of a list of operators $*(0..n)$, as well as the common attributes shown in Subsec. 4.1.1.
Operators are entities responsible for performing a specific task, which may invoke technical services and other infrastructure components. They can be human agents performing human tasks, or software or hardware agents performing automatic tasks.

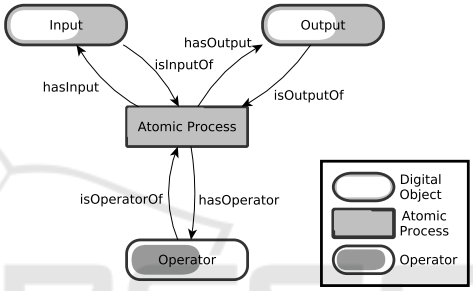

Figure 3: Atomic process and its dependencies with other entities in the ecosystem.

4.1.3 Aggregated Process Attributes

Aggregated processes are created as a combination of other process entities.Thus, they have as additional attributes the description of this combination:
Process flow as an attribute consists of a list of process identifiers that determines the order of execution of sub-processes: a sequential process thread starting with a start event followed by the sub-processes and finishing with an end event.
Data flow as an attribute consists of a map that determines the data connection between input and output slots of process entities and the available digital resources at that time.

4.2 Implementation Entity

An implementation entity contains the information regarding the nature and location of a specific artefact that performs changes on data and systems in the ecosystem. The implementation entity as low-level description of a process consists of a unique identifier and a version number that allows a co-existence of different-aged implementations. Furthermore, it consists of:

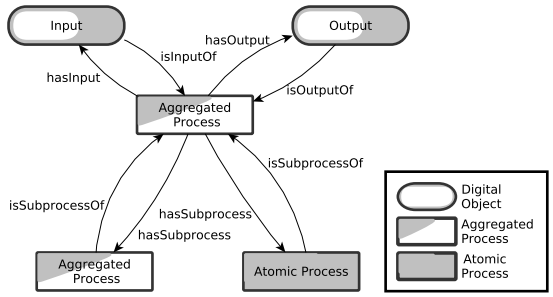

Figure 4: Aggregated process and its dependencies with other entities in the ecosystem.

Implementation Type is an attribute that specifies the identifier of the execution environment, i.e. `BPMN` if the implementation is done as an executable BPMN workflow.
Location specifies where the artefacts containing the implementation are stored. The location annotation is done abstract from real path descriptions as Universal Resource Identifier (URI).
Checksum assures the integrity of the implementation artefact, by providing a cryptographic hash.

4.3 Process Aggregation

To allow a coherent process aggregation approach, we assume the following:
Sequential Execution of the processes that conform an aggregated process. This means that a process can be executed only if the previous one has already finished. Therefore, resources created by previous processes (intermediate resources in Fig. 2) or input resources of the aggregated process can be used as inputs for the current process. Parallel execution in atomic processes is allowed.
Single Threading in aggregated processes. The arrangement of processes is limited to a flat sequence of processes, understanding as a flat sequence a unique thread with a start event followed by specific processes and an end event, (process flow in Fig. 2). Multithreading in atomic processes is allowed.
Immutable Resources, therefore if a process modifies a resource, it has to create a new one. They are represented by files on the file system or arbitrary URIs.
Class based-inheritance in the ecosystem. If an entity, *child* entity, is a specialization of other entity, *parent* entity, then all the attributes of the *parent* entity are present and unmodified in the *child* entity. Thus, we can use a *child* entity as input/output for a process that takes the *parent* entity as the corresponding input/output.

To facilitate the creation of new preservation actions by process aggregation, a set of generic processes that perform common functionalities is provided when creating the preservation system. This set will increase and become richer as new processes are created.

An aggregated process can be described as a combination of atomic and/or other aggregated processes, which can be also described as a combination of atomic and/or other aggregated processes and so on. This *aggregation hierarchy* follows a tree structure in the ecosystem model which always ends in atomic processes with their corresponding operators. Therefore, it is not necessary to explicitly link the list of operators involve in an aggregated process, as they can be inferred by unfolding the process flow steps.

The unfolding step queries extensively the model repository (c.f. Fig. 1), a triplestore which contains the RDF-based descriptions of models and entities, and also integrates a step of verifying data types of input and output slots. Repetitive steps are integrated as multiple occurrences in the process. Due to these constraints of unfolding and assembling processes, the Process Compiler for MDP constructed in PERICLES does not feature a Turing-complete language, but rather gain full explicit process step definitions, important for long-living workflow definitions in the context of data curation. However, process implementations can contain Turing-complete languages with loops and nested instructions. Hence, each step of an aggregated process represents the execution of a process with a well-defined data set.

5 CASE STUDY: INGEST OF THE DIGITAL COMPONENT OF A SOFTWARE-BASED ARTWORK

In PERICLES, the first project stream deals with preservation strategies for digital artworks and media at TATE, as shown in (Falcao et al., 2014). The second stream deals with experimental scientific data originated from the European Space Agency and International Space Station (Soille et al., 2014). Both use case providers are working to understand the impact of the MDP approach on their preservation practices.

In this case study, we focus on the first project stream and its goal is to create a process that ingests an Artwork's Software (AS) into a preservation package of a specific format. We assume that this process can be described as a combination of three processes: check the AS for viruses, extract its meta-

data (MD), and encapsulate the AS together with its MD in a preservation package, ready for storage (c.f. Fig. 6). For the purpose of this paper, we have simplified the ingest process as we only want to depict the process aggregation and show how it facilitates the change management in our MDP approach. A real scenario should consider a more complex design, i.e. the encapsulation process would require stringent quality control to ensure that the performance of the encapsulated software is the same as that of the non-encapsulated version.

Let us consider the following situation: an organisation has a collection of ASs and decides to introduce them in its preservation system. For that, new **policies** and **preservation actions** need to be created and applied, i.e. the policy *"every AS needs to be ingested before storing"*, which implies to execute the process *Ingest AS* over each new AS. A data curator accesses to the preservation system to create the new ingest process and realises that it can be done as a combination of other processes already available in the system. Therefore, the curator decides to create an aggregated process with a process flow composed by: Virus Check, Extract MD and Encapsulate DO and MD processes. These processes are modelled in the ecosystem as atomic process entities with their corresponding operators. The relationships between these processes and other entities in the ecosystem are shown in Fig. 5.

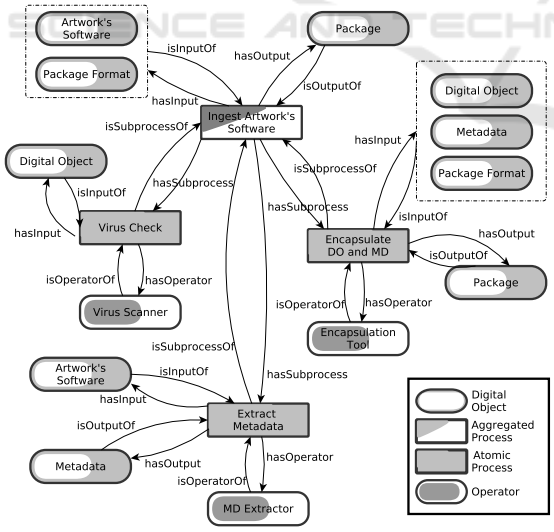

Figure 5: Entities and their dependencies in the ecosystem for the aggregated process *Ingest Artwork's Software*.

Then, the curator has to create the data flow. The *Ingest AS process* has two input resources, an AS and a package format corresponding to the input slots x_1 and x_2 respectively and one output resource, a preservation package corresponding to the output slot y_1 (c.f. Fig. 6).

The *Virus Check process* takes as input the digital material to be checked and produces a file containing the result of this operation. Therefore, the curator maps its input slot with the input slot of the Ingest process related to the AS, that is (x_1, a_1). The result of this check is then used as extra information to be added to the MD of the AS, that is (b_1, c_1). The *Encapsulate digital object (DO) and MD process* encapsulates the AS and its MD together into a package of a specific format. Therefore, the curator maps their input slots with the input slots of the Ingest process, and with the output slot of the *Extract MD* process, that is (d_1, e_3), (x_1, e_2) and (x_2, e_1). The curator also maps its output slot with the output slot of the Ingest process, that is (f_1, y_1). Once the aggregated process is defined, it is compiled by the Process Compiler compiles to create the executable workflow (i.e. BPMN file) to be executed when a new AS is introduced in the preservation system.

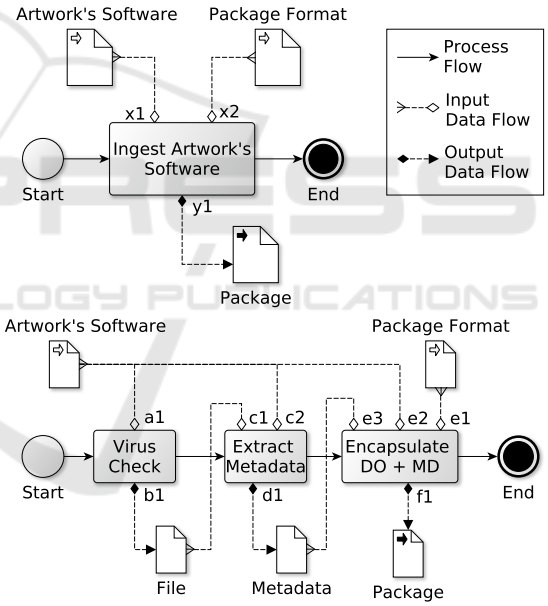

Figure 6: Process flow and data flow for the aggregated process *Ingest Artwork's Software*.

In this example the preservation action is created by hand, but the MDP approach in PERICLES is designed to allow (semi-)automatic reconfiguration of preservation actions. For example, a new preservation policy is applied: *"when checking for viruses, suspect material has to be quarantined for 30 days"*. Therefore, the old process *Virus Check* is changed by a new one *Virus Check and Quarantine*, and all aggregated processes that perform virus checks are automatically reconfigured with the new one.

6 IMPLEMENTATION

The implementation of the PERICLES Process Compiler is done in Java. It currently supports MDP process compilation for BPMN-based workflows, although the design is intended to adapt to other workflow languages. In PERICLES, entities and models are stored in the model repository (c.f. Fig. 1), a triplestore used as database for storage and retrieval of triples through semantic queries. Digital resources and bitstreams are stored in external data storage systems, such as CDSTAR (Schmitt et al., 2014). Communications are done via Representational State Transfer (REST) APIs, allowing in the case of the triplestore SPARQL requests.

The Process Compiler compiles an RDF-based description of an aggregated process to create an executable BPMN file by doing the following:

Validation by validating the file structure against the implementation specification, rejecting input with unknown or unsupported attributes and checking entity compatibility, which may imply to check the inheritance tree and dependencies in the ecosystem model. For instance, the Virus Check process in Sec. 5 takes as input an entity of class *Artwork's Software*. As the AS class is a subclass of *Digital Object*, this process can be used to check an entity of class AS and the previous mapping (x_1, a_1) is valid.

Process Flow Handling by linking the processes in an appropriate order taking into account the services used to perform the tasks and managing the corresponding interfaces. It creates a single thread with start-event the one of the first sub-process and end-event the one of the last sub-process.

Data Flow Handling by connecting input and output process slots with external and temporary digital resources according to the map specified by the data flow of the aggregated process. Intermediate resources are temporarily stored in a secure external system (i.e. CDSTAR) and deleted when the process compilation is finished.

Error Handling by generating appropriate errors and linking them to external events.

7 CONCLUSION

The PERICLES project follows a model-driven preservation approach, where the digital ecosystem is modelled and updated to handle changes in digital information and mitigate their impact for preservation purposes. The *PERICLES Process Compiler* supports this model-driven approach by validating and linking processes to perform preservation actions over evolving ecosystems.

For doing this, we introduce the concept of aggregated and atomic processes as part of the ecosystem model, which are used by the Process Compiler to compile preservation processes into executable BPMN workflows. Using these different process types allows more flexibility to create new preservation processes, increases process reusability, and reduces the scope of changes in the ecosystem. This approach provides as well the advantages of semantic annotation of processes, namely better process understanding and representation, and process query and validation at semantic level, without developing an entire complex process ontology. Also, the relationships and dependencies between the entities in the ecosystem become clearer as low-level descriptions remain hidden in implementation entities linked to process entities. Starting with a set of common processes, preservation processes become more complex and specialised as the ecosystem evolves. The Process Compiler facilitates the creation and modification of these processes without a deep knowledge of the implementation language or the ontologies and models used for describing the ecosystem.

The Process Compiler is still under development and its functionality is subject of research in the course of the PERICLES project. Current scenarios in both use cases look at the Process Compiler as a key component to perform the MDP approach, as it allows the automatic reconfiguration of preservation actions when a change occurs. For instance, changes in a policy imply the reconfiguration of its associated processes to be applied over the digital ecosystem to assure its coherent state, or changes in technology, i.e. obsolete software, imply the recompilation of the implementation files to be linked to up-to-date services.

8 FUTURE WORK

Future lines of work of the PERICLES Process Compiler will allow more complex process aggregations, not only sequential threads with input and output mapping, but actual complete automata by leveraging Petry nets. On the other hand, semantic validation will be added to type validation together with optimizations at some degree, i.e. verify that a file is checked for viruses only once, to improve the performance of the overall system where the Process Compiler is used. Although BPMN is a widely used process modelling standard, another interesting feature is to support other workflow description languages in order to extend its usability and allow process models interoperability.

ACKNOWLEDGEMENTS

Parts of this work are founded in the PERICLES project funded by the European Union under its Seventh Framework Programme (ICT Call 9) under the grant agreements No 6 01138. We thank Marcel Hellkamp for his contribution on the Process Compiler implementation and sharing his ideas and concepts with us. Furthermore, we like to thank Patricia Falcao from TATE for giving insights into their use cases.

REFERENCES

Barateiro, J., Draws, D., Neuman, M., and Strodl, S. (2012). Digital Preservation Challenges on Software Life Cycle. In *Software Maintenance and Reengineering (CSMR), 2012 16th European Conference on*, pages 487–490, Szeged.

Binz, T., Breiter, G., Leyman, F., and Spatzier, T. (2012). Portable Cloud Services Using TOSCA. *IEEE Internet Computing*, 16(3):80–85.

Brocks, H., Kranstedt, A., Jäschke, G., and Hemmje, M. (2010). Modeling Context for Digital Preservation. In Kacprzyk, J., Szczerbicki, E., and Nguyen, N. T., editors, *Smart Information and Knowledge Management*, volume 260, pages 197–226. Springer, Berlin.

Davenport, E. and Cronin, B. (2000). Knowledge Management: Semantic Drift or Conceptual Shift? *Journal of Education for Library and Information Science*, 41(4):294.

Falcao, P., Ashe, A., and Jones, B. (2014). Virtualisation as a Tool for the Conservation of Software-Based Artworks - Presentation. In *Proceedings of the 11th International Conference on Digital Preservation*, pages 83–90, Melbourne. State Library of Victoria.

Glaser, F. (2015). Towards Domain Model Optimized Deployment and Execution of Scientific Applications in Cloud Environments. In *Doctoral Consortium*, pages 20–25, Lisbon, Portugal.

Gottschalk, F., Van Der Aalst, W. M., Jansen-Vullers, M. H., and La Rosa, M. (2008). Configurable workflow models. *International Journal of Cooperative Information Systems*, 17(02):177–221.

Höfferer, P. (2007). Achieving Business Process Model Interoperability Using Metamodels and Ontologies. In *ECIS*, pages 1620–1631, Geneva, Switzerland.

Koliadis, G., Vranesevic, A., Bhuiyan, M., Krishna, A., and Ghose, A. (2006). Combining i* and BPMN for Business Process Model Lifecycle Management. In Hutchison, D. and Eder, J., editors, *Business Process Management Workshops*, volume 4103, pages 416–427. Springer, Berlin.

Kowalczyk, S. (2008). Digital Preservation by Design. In Raisinghani, M. S., editor, *Handbook of research on global information technology management in the digital economy*, pages 405–431. Information Science Reference, Hershey.

Ludäscher, B., Altintas, I., and Gupta, A. (2003). Compiling abstract scientific workflows into web service workflows. In *Scientific and Statistical Database Management, 2003. 15th International Conference on*, pages 251–254, San Diego. IEEE.

Martin, D., Paolucci, M., McIlraith, S., Burstein, M., McDermott, D., McGuinness, D., Parsia, B., Payne, T., Sabou, M., Solanki, M., Srinivasan, N., and Sycara, K. (2005). Bringing Semantics to Web Services: The OWL-S Approach. In Hutchison, D., editor, *Semantic Web Services and Web Process Composition*, volume 3387, pages 26–42. Springer, Berlin.

Matthews, B., Shaon, A., Bicarregui, J., and Jones, C. (2010). A Framework for Software Preservation. *International Journal of Digital Curation*, 5(1):91–105.

Neumann, M. A., Riedel, T., Taylor, P., Schmidtke, H. R., and Beigl, M. (2011). Monitoring for Digital Preservation of Processes. pages 214–220, Berlin. Springer Berlin Heidelberg. Last accessed on Oct 27, 2015.

PERICLES, P. (2015). About the project. Last accessed on Oct 27, 2015.

Rao, J., Kungas, P., and Matskin, M. (2004). Logic-based web services composition: From service description to process model. In *Web Services, 2004. Proceedings. IEEE International Conference on*, pages 446–453, San Diego. IEEE.

Rao, J. and Su, X. (2005). A Survey of Automated Web Service Composition Methods. In Hutchison, D., editor, *Semantic Web Services and Web Process Composition*, volume 3387, pages 43–54. Springer, Berlin.

Schmitt, O., Siemon, A., Schwardmann, U., and Hellkamp, M. (2014). GWDG Object Storage and Solution for Research - Common Data Storage Architecture (CDSTAR). *GWDG Berichte*, (78):64. Last accessed on Oct 27, 2015.

Sirin, E., Hendler, J., and Parsia, B. (2002). Semi-automatic Composition of Web Services using Semantic Descriptions. In *In Web Services: Modeling, Architecture and Infrastructure workshop in ICEIS 2003*, pages 17–24, Angers, France.

Smith, F., Missikoff, M., and Proietti, M. (2012). Ontology-Based Querying of Composite Services. In Hutchison, D., editor, *Business System Management and Engineering*, volume 7350, pages 159–180. Springer, Berlin.

Soille, P., Marchetti, P. G., European Commission, Joint Research Centre, Institute for the Protection and the Security of the Citizen, ESA, SatCen, and Conference on Big Data from Space (2014). *Proceedings of the 2014 conference on Big Data from Space (BiDS'14)*. Publications Office, Luxembourg.

Thomas, O. and Fellmann, M. (2007). Semantic EPC: Enhancing Process Modeling Using Ontology Languages. In Hepp, M., Hinkelmann, K., Karagiannis, D., Klein, R., and Stojanovic, N., editors, *Proceedings of the Workshop on Semantic Business Process and Product Lifecycle Management (SBPM 2007)*, Aachen. RWTH Aachen.

Vion-Dury, J.-Y. (2015). Towards model-driven preservation. Colmar, France.

Vion-Dury, J.-Y., Lagos, N., Kontopoulos, E., Riga, M., Mitzias, P., Meditskos, G., Waddington, S., Laurenson, P., and Kompatsiaris, I. (2015). Designing for Inconsistency – The Dependency-Based PERICLES Approach. In Morzy, T., editor, *New Trends in Databases and Information Systems*, volume 539, pages 458–467. Springer International Publishing, Cham. Last accessed on Oct 27, 2015.