Towards a Decision Support Architecture for Digital Preservation of Business Processes

Martin A. Neumann
KIT, TECO
Karlsruhe, Germany
mneumann@teco.edu

Hossein Miri
KIT, TECO
Karlsruhe, Germany
miri@teco.edu

John Thomson
Caixa Magica Software
Lisbon, Portugal
john.thomson@caixamagica.pt

Goncalo Antunes
INESC ID
Lisbon, Portugal
goncalo.antunes@ist.utl.pt

Rudolf Mayer
Secure Business Austria
Vienna, Austria
mayer@sba-research.at

Michael Beigl
KIT, TECO
Karlsruhe, Germany
beigl@teco.edu

ABSTRACT

In this paper, we present and address a number of challenges in digital preservation of entire business processes: (1) identifying digital objects a business process depends on ("What to preserve and why?"); (2) identifying significant changes in digital objects ("When to preserve?"); (3) determining a re-deployment setting ("What to re-deploy and why?"). After highlighting these challenges, we illustrate some aspects of business processes that are relevant in the context of digital preservation and provide a model to capture their semantics formally. We, then, proceed to present a decision support architecture and address the challenges using the developed model, as well as pointing out some of its limitations. We, finally, conclude the paper by discussing the applicability of our proposed techniques and model.

Keywords

Digital Preservation, Decision Support, Business Processes

1. INTRODUCTION

Digital preservation research is concerned with providing long-term *access to* and *intelligibility of digital objects*, regardless of their complexity. It focuses on preserving digital objects along with their meta-data (or contextual information) required to achieve this goal [5]. Digital preservation research has so far focused on digital objects which are static in nature, meaning they do not perform active behavior over time. In digital preservation communities, such as libraries and national archives, this includes text and multimedia documents.

Today however, an increasing amount of static digital objects are replaced by dynamic ones—e.g. dynamic websites, results of e-science experiments, generated meta-data, etc. This dynamic content is generated using processes such as the simplified *automated documents classification process* depicted in Figure 1. This means that to preserve the entire scope of digital objects, the processes that define the context (within which objects are accessed and interpreted) have to be preserved as well.

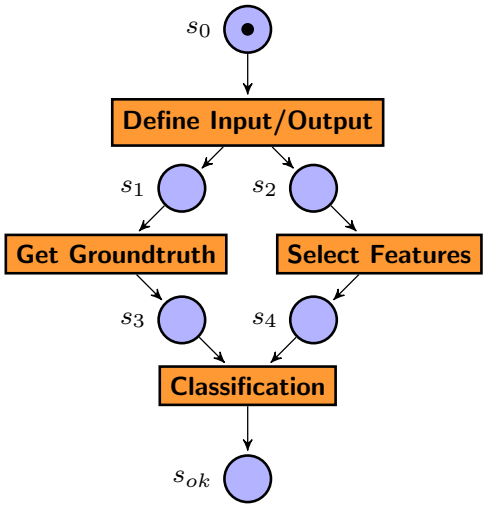

Figure 1: Classification Process to be Preserved

For this reason, recent research activities have focused on extending established preservation approaches to dynamic digital objects; referring to those that actively perform behavior over time. *Active behavior* describes any externally-visible actions performed by the digital object to interact with its environment. It also refers to any actions performed purely internally which are not externally visible. Video games and computing systems in general are well-known examples of such dynamic digital objects [11]. In this paper, the primary focus is on distributed and networked computing systems, implementing processes that drive businesses.

It is also noteworthy that, as pointed out in [12], modern business processes comprise complex ecosystems. A process may span many involved legal parties and may be supported by a heterogeneous, distributed, technical infrastruc-

ture. The infrastructure's components are technically homogenized and enabled for remote integration using service-oriented middleware (also called service-oriented architecture). Furthermore, they may be integrated into business processes using service delivery concepts, such as IoS (Internet of Services) and {Software, Platform, Infrastructure, ...} as a Service.

We, therefore, present and address here a number of challenges in digital preservation of entire business processes that have been identified in the context of a digital preservation project that focuses on time-resilient business processes. The procedures for the preservation of whole business processes are: (a) preservation planning, (b) preservation execution, and (c) preservation re-deployment (exhumation of a preserved process) of a preserved process. In the context of these procedures, relevant challenges are:

What to preserve and why? During preservation planning, we identify *what* digital objects a business process depends on and *why*.

When to preserve and why? During preservation planning, we identify relevant differences in digital objects to detect *when* to preserve a business process and *why*.

What to re-deploy and why? Before re-deployment, we identify *what* suitable re-deployment settings, in terms of *what* preserved digital objects will be re-deployed in *which* re-deployment environments[1].

In Section 1.1, we discuss the context of business processes relevant to digital preservation and how to model it. In Section 1.2 we discuss how to establish decision support for digital preservation activities based on this model.

In Section 1.3, we point out three reasoning tasks in the context of preservation planning, execution and re-deployment for business processes. In order to define the scope of this paper, we only focus on these three tasks (which are closely related and involve the entire preservation process). Section 2 illustrates the proposed model that has been developed for the digital preservation of business processes which will be further revised. This model captures knowledge which is generally relevant to digital preservation of business processes, based on a set of representative use-cases and an enterprise modeling framework.

In Section 3, we explain how we address the reasoning tasks based on our model and a proposed decision support architecture. We also analyze the computational complexity of our three proposed approaches. Finally, we discuss the applicability of our approach to preservation of business processes, and then conclude in Section 4.

[1]An adjustable part of a re-deployment environment may be adapted during the re-deployment procedure to accommodate for the process-specific situation established by the preserved digital objects and a partially fixed re-deployment environment.

1.1 Relevant Context of Business Processes

We argue that there are many aspects in the context of a business process that have to be taken into account during preservation planning and execution to ensure successful re-deployment of that process. We consider "successful re-deployment" as the ability to re-run a preserved process which behaves in the same way as the original one[2] [1]. Additionally, we argue that, in the context of a business process, (1) there are abstract (coarse-granular) aspects which are relevant to the entire domain of process preservation, and (2) there are more specific aspects (fine-granular) which are relevant to sub-domains of process preservation, e.g. the *class of scientific processes* or an *individual scientific experiment*, which may identify further relevant aspects. For example, at the most coarse-granular level, we have identified the following abstract categories of aspects as being relevant to the entire domain of business process preservation: (1) processes, (2) preservation requirements, (3) services, (4) software, (5) hardware, (6) licenses, (7) authorizations, (8) people. The elements of these categories combine to form a complex inter-dependent network of different types of classes, individuals, relations and rules—they form an upper ontology capturing the knowledge relevant to business process preservation in general. This ontology may be lowered to sub-domain- or even process-specific ones to capture the knowledge relevant to the respective sub-domain.

In terms of decision support for preservation activities, there is an issue of these aspects forming large networks. Conceptually, we can use these networks of aspects to assist in activities by drawing conclusions from them, as illustrated in Section 1.3. However, the networks' complexities could hinder digital preservation engineers from sketching them on a blackboard and manually drawing conclusions. If we model these aspects and their inter-relations semantically adequately, we can support planning, execution and re-deployment activities using reasoning on these models. *Semantically adequately modeled* means that the model captures the semantics of the business process and its context in such a way that is suitable for automatically drawing conclusions that are of practical use for process preservation. The practical suitability of our model and results derived by reasoning on it have to be experimentally evaluated.

There are several models in the literature that capture the context relevant to digital preservation of digital objects—often referred to as *representation information*. Prominent examples are the PREMIS data dictionary for preservation meta-data [14], and representation information networks [9]. In this paper, *context relevant to digital preservation* and *representation information* both refer to information that a designated user community requires to comprehend the preserved digital objects properly— i.e. intelligibility of digital objects to a designated group of people at some future point in time [5]. These models focus on "structural aspects" of digital objects (i.e. mereological relationships), and do not take "behavioral aspects" into account [11]. *Structural as-*

[2]It behaves equivalent according to an equivalence notion, such as trace equivalence [13], and equivalent in terms of relevant modalities, such as causality and time. Both aspects are determined by the requirements of process preservation in general, but also by the requirements of preserving the process in focus.

pects reflect the idea of representation information: capturing the necessary contextual information such that a designated user community can understand the preserved digital objects. From our perspective, executional aspects are relevant, because we have to model systems which are complex objects on the one hand (as business processes have a compositional structure of inter-related parts), and those which perform actions (behave) on the other hand. Thus, in addition to a structural notion and model, we need a notion and model of behavior which is adequately applicable to digital preservation of business processes. As stated before, this notion and model of behavior has to accomplish the above goal of enabling successful re-redeployment of a preserved process. As a consequence, we extend the interpretation of the term "digital preservation relevant context" in the context of business processes to: referring to information that a designated user community requires to comprehend archived digital objects properly, as well as information that a designated user community requires to validate the execution of a re-deployed behavioral system. We also propose a novel modeling approach for digital preservation of business processes that captures relevant structural and behavioral aspects to enable successful re-redeployment of a preserved process. However, as mentioned above, whether the modeling approach achieves this goal has yet to be evaluated in representative case studies of process preservation.

1.2 Decision Support for Digital Preservation

Previous models used for capturing the context relevant to digital objects focus on semantically-restricted ontologies, in terms of offering only a handful of types of: (1) classes of digital objects, (2) classes of things in the context of digital objects, (3) and relations between these classes. In addition, they capture little additional background knowledge on these classes and relations, especially in terms of rules or rule-style statements. Consequently, drawing conclusions from the modeled knowledge is limited without the interpreter having exhaustive background knowledge. This background knowledge would enable the interpreter to "semantically richly" interpret a given model, and thereby draw conclusions from it which would go beyond the knowledge captured in the model. For example, an interpreter could know that there is a licensing issue in re-running a preserved operating system (which is also captured in the model) based on personal experience only.

The PREMIS data dictionary is special in this regard, as it covers only a handful of types of classes and relations between them, but it specifies a large amount of relations between objects and primitive data types (which are called "semantic units"). Some semantic units are mandatory, some are optional and in general they can be used to add a lot of preservation relevant detail to the modeled objects, for example, but not exclusively digital objects that are to be preserved. Our modeling approach uses the semantic units specified for digital objects in PREMIS: transferring a lot of semantic detail from mature digital preservation modeling research, and making our model partially PREMIS-compatible. However, unlike our approach, the PREMIS data dictionary aims for intelligibility, and it does not cover semantics to derive defined conclusions from the model which could, for example, assist in decision support of preservation activities.

Besides the captured behavioral aspects, our modeling approach captures the introduced "structural aspects" on a "semantically rich" level (ontology in general) that has not been provided by previous modeling approaches. This has two advantages: (1) the comprehensibility of preserved digital objects is improved without the need for the interpreter to have exhaustive background knowledge[3], and (2) reasoners that assist during preservation planning, execution and re-deployment can directly operate on the knowledge kept with a preserved digital object. Furthermore, the knowledge kept with a preserved digital object can even be specific to this digital object, which means (in our case) that the model is specific to the preserved business process. A reasoner would directly be able to draw conclusions from it without having to combine the knowledge kept with the digital object with the background knowledge kept inside the reasoner itself. Combining both would be necessary, if the reasoner would bring in knowledge in addition to the knowledge kept with a digital object. In this case, both knowledge bases are in danger of contradicting each other and, therefore, hard to combine [3]—in particular, if both knowledge bases originate from different contexts, such as points in time, or user communities. This implies another positive aspect of the second advantage of our approach: in general, reasoners do not have to be sub-domain- or process-specifically adapted and are time-resilient.

As mentioned, we promote the use of an ontology to model knowledge on digital objects, and also to design digital object-specific models to accommodate for specific requirements on digital preservation of an object. For example, in one scenario it might be fine to re-deploy a business process which exposes causal trace equivalent behavior after re-deployment. However, in the case of a scientific experiment, causality and exact timing are likely to be very relevant. Therefore, if we would, for example, like to assist preservation planning in answering the question "what to preserve?" for both processes, there is no generic strategy to answer it. For the first process, it could be sufficient to only preserve technical requirements down to the operating systems which in this example are known to provide a run-time environment that preserves causality. In the case of the second process, we might need to preserve technical requirements down to the hardware, which is assumed to provide cycle-time accurate timing. Therefore here, we need two different strategies (or policies) to determine which parts of the business processes are required to be captured—the strategy is specific to the digital object in focus, and thus has to be kept with object itself and not the reasoner.

We envision that many digital preservation-related questions are specific to digital objects, analogous to the illustrated example. The preservation questions depend on the context (or situation). Therefore, we argue that it is important to provide the ability to capture digital object-specific knowledge for their digital preservation, in particular, for business processes: to improve the understanding of preserved digital objects without the need for background knowledge, and also to enable generic reasoning mechanisms to act on the preserved digital object only to assist in preservation activities, such as planning, execution and re-deployment.

[3]For example, one book on the used preservation technology and another one on the preserved business process.

In the past, digital preservation research has already implemented decision support approaches—the most recent one is Plato [2]. In contrast to our methodology, Plato focuses on digital objects which are static in nature, and as such do not perform active behavior over time. For example, text documents or images. Plato provides a reasoning framework for identifying actions to preserve a digital object. In general, this idea complements the approach pursued in this paper, as we do not discuss the question of "how to preserve a digital object?". As we are concerned with dynamic digital objects, Plato's applicability to this new domain will have to be evaluated in the future. To achieve its goal, Plato (1) defines generic features of digital objects, such as the presence of intellectual property rights issues; (2) defines more specific features of classes of digital objects, such as compression characteristics of image formats; (3) devises methods to extract these features from digital objects, such as by using tools or performing manual experiments; and (4) proposes a reasoning method to conclude optimal preservation actions from the features of a digital object. This methodology is in line with our vision and requirement of being able to draw conclusions from the model of a digital object only. This is due to the fact that a generic reasoning mechanism is proposed that calculates and compares the "utilities" of preservation actions on a unified scale, whereby the feature extraction techniques of a digital object are responsible for providing a strategy to map their outputs onto this scale. In the worst case, each digital object would take along its specific feature extraction techniques. Our future research efforts will investigate integrating both approaches.

1.3 Process Preservation Challenges

In order to be correctly rendered, a digital object needs a technological context resulting from the combination of specific hardware and software. Moreover, in order to be correctly understood by humans, the organizational/business/social context surrounding the object is also needed. The Digital Preservation Europe research roadmap, published in 2007, defines the context of a digital object as the "representation of known properties associated with and the operations that have been carried out on it" [6]. On the one hand, the aforementioned properties might include information about the technology used, but on the other hand, those properties might consist of legal requirements, existing knowledge, and user requirements. The operations performed on an object might include the processes that originated the object itself.

The determination of the relevant context of a digital object becomes even more challenging if complex digital objects such as workflow or business process specifications are considered. Those types of objects are dependent on an highly complex and distributed technical infrastructure hosted in complex and diverse organizational settings, sometimes involving multiple organizations. This creates a complex dependency network involving the object and other complex objects on which its correct rendering and understanding depends upon. However, not all context might be relevant for being able to correctly preserve and redeploy a process in the future. Some of the context information might not even be available at all. In fact, a selective approach for determining the context of a process should be pursued in terms of the usage of the model. Otherwise, it might lead to resource waste, and it might even cause the costs of preservation to surpass its potential benefits. In that sense, the first preservation challenge faced when dealing with the preservation of business processes is "what to preserve and why?".

After the identification of the relevant contextual information it becomes necessary to determine how to approach the capturing and preservation of the process and relevant context information. In other words, it is important to determine what and when preservation actions will be processed. As introduced, this issue has so far been addressed by Plato and is therefore out of the scope of this paper. It is assumed that surpassing this challenge will result in the successful execution of the preservation actions that will allow the process and its relevant context to be preserved.

However, the fact that we are dealing with complex and dynamic objects leads to new challenges. Despite the fact that a specific instance of the process and its context is captured at a determined time while undergoing preservation, it becomes crucial to monitor the original process as currently deployed in the original setting to detect any changes that might occur. Since those changes might be potentially important to capture, another preservation challenge being faced is "when to preserve and why?". Facing this challenge successfully will involve having several snapshots of the process and its relevant context documenting the main events happening during the life-cycle of the process.

Different challenges can also be faced during the re-deployment of a preserved process. Since digital preservation concerns the long-term, it is highly probable that the original deployment setting are partly or not-available at all. The context model along with the preserved context information should provide information on what are the suitable re-deployment settings for the preserved processes. The redeployed environment might need adaptation during the redeployment procedure in order to replay any situation of interest. Hence, a challenge that must be faced in the redeployment of business processes includes knowing "what to re-deploy and why?".

After the identification of the relevant context elements that are absolutely needed for the correct re-deployment of a business process, it becomes necessary to determine how to approach the re-deployment itself. So it becomes crucial to determine what and when re-deployment actions will be processed. This issue is quite analogous to what is being addressed by Plato and is therefore too out of the scope of this paper. This issue will be surpassed if the re-deployment of the process and environment allow for the correct and successful re-execution of the process. Which is an issue we are trying to solve using monitoring of a re-deployed process to verify its behavior, as presented in [12]. It covers an approach to monitoring of business processes to trigger their digital preservation and verifying their causal behavior based on a notion of trace equivalence. In the context of the challenge "when to preserve and why?", this paper will extend these findings by a notion of trace equivalence to verify the causal and temporal behavior of processes.

2. CONTEXT MODEL

Our *context model* describes business processes and their context, both of which are scoped to aspects relevant to the

digital preservation of the processes. A model $\mathcal{M} := \langle \mathcal{B}, \mathcal{C} \rangle$ consists of a set of business processes $\mathcal{B}$ and a context $\mathcal{C}$.

As introduced in [12], we have identified *condition/event structures* (or 1-safe petri nets) as being an adequate notion for modeling the structure and causal behavior of business processes. It is an approach for design and *efficient* verification which clearly formulates causal behavior of concurrent systems [4]. To be able to additionally model temporal behavior of business processes, as required in this work, we extend our notion to *time condition/event structures*. This approach allows to model *causal and temporal behavior* of concurrent processes for design and verification.

A condition/event structure $\mathcal{N}^{c/e} := \langle \mathcal{P}, \mathcal{T}, \mathcal{F}, m_0 \rangle$ consists of a set of *places* $\mathcal{P}$ encoding *conditions* and a set $\mathcal{T}$ of *transitions* encoding *events*, where $\mathcal{F} \subseteq (\mathcal{P} \times \mathcal{T}) \cup (\mathcal{T} \times \mathcal{P})$ is the set of edges of the net and m_0 is the *initial marking*. Here, a function $m_i : P \to \{0, 1\}$ is called a *marking*. A transition t is activated ("may fire") in a marking m_i iff for all p holds: (1) if $(t, p) \in \mathcal{F}$ then $m_i(p) = 0$, and (2) if $(p, t) \in \mathcal{F}$ then $m_i(p) = 1$. A sequence of "fired" transitions $t_i \to \ldots \to t_j$ is called a *trace*.

A time condition/event structure $\mathcal{N}^{t,c/e} := \langle \mathcal{P}, \mathcal{T}, \mathcal{F}, m_0, l \rangle$ consists of a condition/event structure $\langle \mathcal{P}, \mathcal{T}, \mathcal{F}, m_0 \rangle$ and a *time labeling function* $l : \mathcal{T} \to \mathbb{N}_{\geq 0} \times \mathbb{N}_{\geq 0} \cup \{\infty\}$ whereby for all $t = (t_i^\circ, t_i^\bullet)$ holds: $t_i^\circ \leq t_i^\bullet$ and $t_i^\bullet < \infty$. All t_i° are called *earliest firing times* and all $t_i^\bullet$ are called *latest firing times*. A transition "may fire" the earliest at its t_i° and "has to fire" the latest at its $t_i^\bullet$ since its activation. Furthermore, $j_i : \mathcal{T} \to \mathbb{N}_{\geq 0} \cup \{\phi\}$ is a *clock function* that gives the time which has elapsed since a transition t has been activated. In consequence, for all t_i holds: $j_j(t_i) \geq t_i^\circ$ and $j_j(t_i) \leq t_i^\bullet$. A sequence of time-annotated "fired" transitions $(t_i, j_i) \to \ldots \to (t_j, j_j)$ is called a *time trace*.

Now, the set of business processes $\mathcal{B}$ in our model can be defined as a set of time condition/event structures: $\mathcal{N}_i^{t,c/e} \in \mathcal{B}$. Furthermore, the context $\mathcal{C} := \langle \mathcal{E}, \mathcal{R}, \mathcal{S} \rangle$ consists of a set of classes $\mathcal{B}$, a set of relations $\mathcal{R}$ and, a set of logical statements $\mathcal{S}$. Each class $b_i := \{i_0 \ldots i_n\}$ is a set of individuals i_j. Each relation $r_i \subseteq (\mathcal{T} \times \mathcal{E}) \cup (\mathcal{E} \times \mathcal{E})$ relates transitions (i.e. events) to classes, and classes to classes. Each logical statement s_i is a horn-formula in predicate logic[10] whereby its predicates are restricted to the relations in $\mathcal{E}$ and $\mathcal{R}$.

At its core, our model is a formal ontology framework that can be instantiated for digital preservation settings which involve concrete business processes and their context. Instantiation of the framework involves the definition of classes, individuals, relations and logical statements. This ontology framework provides the ability to model processes and their context in a semantically comparably rich way, as motivated in Section 1. We have investigated what individuals, classes, relations and statements apply to the entire domain of digital preservation of business processes. This specifies and scopes the introduced generic formal ontology framework to the domain of digital preservation of business processes by populating $\mathcal{E}$, $\mathcal{R}$ and $\mathcal{S}$ by domain-specific details. Our design methodology of our domain-specific ontology has already been presented in [11], but we introduce it briefly in the to sketch how the eight abstract categories of aspects presented in Section 1.1 have made their way into the ontology, but we leave the actual details of the ontology for future publication.

In terms of modeling the domain-specific ontology, we adopted a similar approach to the middle-out approach suggested in [17] that has been used in methodologies for designing ontologies such as in [7]. For the top-down, hierarchical view and for dividing the enterprise we have chosen the Zachman framework [18] as a well tried and tested approach for dividing the concerns that is understandable and whose ideas are used currently by many enterprises either directly or indirectly. The Zachman framework guides the division of an enterprise into many perspectives and allows for a holistic view of an enterprise to be captured. Conversely, for the bottom-up approach, a set of partner-specific scenarios formed the foundation. Involved partners, including large international, research and small-to-medium business institutions, came up with different motivational scenarios where digital preservation is important. The scenarios were then analyzed in expert workshops to identify core concepts and relations. The scenarios did not represent the entire breadth of enterprises but provided a step towards identifying our eight different abstract categories of concern. The core aspects were used as the basis for building an initial ontology. Further scenarios are being investigated that will identify new aspects relevant to the entire domain of business process preservation and test-drive already identified ones.

3. ADDRESSING THE CHALLENGES

Figure 2 presents our proposed architecture to provide decision support in terms of the presented challenges. In a concrete digital preservation setting, the context model (1) introduced in Section 2 is firstly fed into the "Model Builder" to create a specialized instance of the model—it ingests our ontology which is specific to the entire domain of process preservation to create an instance of it which is specific to the process. Secondly, to create this instance, relevant knowledge from knowledge bases[4] (2—such as data formats and software licenses) and process-specific details (3—such as process-specific preservation requirements, and involved software and hardware) are added to the ontology by the "Model Builder". The process-specific details may either be automatically extracted from a business process (e.g. software and hardware) or manually input by digital preservation engineers (e.g. preservation requirements).

The produced model (4) captures all knowledge relevant to digital preservation of the process in focus and this model will accompany the process during its entire life-cycle in a preservation archive. Furthermore, the model contains the knowledge required to provide decision support to the three presented preservation challenges, as will be illustrated in the following sections. In general, as our model is based on individuals (objects), classes (unary relations), binary relations and horn formula in first-order logic, the produced model can be handed over to various types of semantic reasoners (such as standard description logic or first-order logic reasoners) to conclude solutions from given problems based on the given model only.

[4]The knowledge bases conceptually are part of the ingested context model, but are kept separate from it in our implementation.

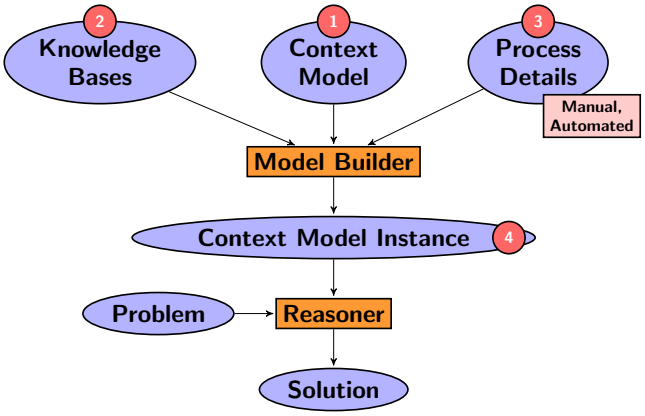

Figure 2: Decision Support Architecture

3.1 Objects to Preserve

As motivated earlier, answering the question of "what to preserve and why?" can be reduced to establishing a notion of what is *required* by a process to be preserved and successfully re-deployed. This notion is determined by preservation requirements which are relevant to the entire domain of process preservation, and more specific requirements which are relevant to sub-domains of process preservation. For example, as illustrated, each process imposes individual requirements on its causality and timing equivalence. Therefore, this notion is specific to the process and the digital preservation setting[5] (called *process-specific* in the following).

In general and in our ontology, there are several ways to model *what is required* by a process. One approach is to explicitly model a *requires relation*. For example, we could say that "a program requires an operating system, which requires a machine, which requires an operator". This is semantically rather shallow, and the covered previous modeling approaches are able to accomplish this. There is no need for our idea of a "semantically rich" context model. But this approach does not provide a process-specific notion of what is required. If we capture a model of only *requires relations*, for example of all software and hardware components involved in a process, we cannot tell what components are "really necessary" for successful re-deployment without manually inspecting and modifying the model, which is likely a labor intensive task and likely to loose information relevant to yet unknown re-deployment settings.

Another approach would be to implicitly model a *requires relation* by declaring other relations, such as *runsOn*, *isInstalledOn*, *isOperatedBy* of being a subtype of the *requires relation*. Based on this, we could process-specifically select which relations determine *what is required*. For example, we could model that "a program isInstalledOn an operating system, which runsOn a machine, which isOperatedBy an operator" and conclude that all four individuals are required by our process. But this is still semantically rather shallow, as we still could not process-specifically distinguish, for example, "really necessary" software and hardware components from "not really necessary" ones.

Therefore, we argue that a more expressive approach is required which provides a more complex notion of *what is required*, and we propose horn formula in predicate logic to express this process-specifically on our ontology. It allows to express that all objects that satisfy a complex statements are required. For example, "it is only necessary to preserve an operating system if it is proprietarily licensed". We are in the process of implementing this approach using the description logic-safe part of the semantic web rule language [8] and the Pellet reasoner [15]. Based on this, all problems given to our reasoning engine are decidable, although the employed language exposes a worst-case computational complexity in reasoning of N3ExpTime (upper bound). Our future efforts will determine which language fragments are required in process preservation practice to improve on the complexity and whether it is a computationally tractable approach.

3.2 Events to Preserve

As motivated earlier and discussed in [12], answering the question of "when to preserve and why?" can be reduced to establishing a notion of what is the *difference* between the process now and when it has been preserved the last time. If this difference exceeds some level of relevance, then a new trigger to preservation execution is determined. Again, this notion of *what a relevant difference in what modalities is*, is process-specific, as each process imposes individual requirements on its causality and timing equivalence.

We propose a notion of *trace equivalence* to detect *relevant differences* in causality and timing behavior of a process at two different times. Our idea is based on the detection of relevant differences in the execution traces of processes under equivalent contextual conditions (regarding their interaction with the environment, such as values of inputs). Based on the *traces* and *time traces* of processes that are defined in our model (in Section 2), we can compare traces stored in two models with each other. Comparing any two traces requires that they have been taken under equivalent contextual conditions—they are called *comparable traces* in the following. We propose a *process-agnostic* notion of difference in the qualitative order of events, and a *process-specific* notion of difference in the quantitative order of events.

Regarding the qualitative difference notion, any change in the qualitative order of events between two comparable traces marks a *relevant difference*. Regarding the quantitative difference notion, *deviations* of an event's timing (in a *time trace*) from its time interval[6] marks a trace which deviates from its process specification. Incorporating the process (of which the trace has been taken) is important in this case, as the quantitative difference notion is process-specific. Two *comparable time traces differ relevantly* from each other, if and only if one of them deviates from the timing interval specification and the other one does not. Each process defines an individual interval of expected timing values for each event, as defined in our model in Section 2. These individual

[5]In this context, the *setting* particularly refers to the temporal preservation horizon which determines setting-specific aspects such as available technologies and relevant user communities of the future.

[6]Time interval specification of the event in the *time condition/event structure* of the process, in our model.

interval information can be either given by expert knowledge or by profiling a process.

The causal and timing behavior of a process, during its execution under specific contextual conditions, is given by one *time trace* in our model. If we want to capture the behavior of a process under varying contextual conditions, we need to capture a set of *time traces*, along with their contextual conditions, in our model. To compare two processes, we compare their trace sets. The trace sets have to have been taken under the same varying conditions. Each two traces that have been taken under the equivalent conditions have to be compared with each other. If this fails on at least one set of two traces, a *relevant difference* has been identified. When this approach is applied to monitoring of a process which is to be preserved, the identified relevant difference represents a trigger ("when to preserve and why?") to preservation of the process.

We are in the process of implementing this approach using the description logic-safe part of the semantic web rule language [8] and the Pellet reasoner [15]. We require a language fragment which exposes a worst-case computational complexity in reasoning of NExpTime (upper bound). Our future efforts will determine if we can restrict the language fragment even further to improve on the complexity, and we will determine whether it is a computationally tractable approach to process preservation practice.

3.3 Objects to Re-Deploy

Although it seems analogous, answering the question of "what to re-deploy and and why?" is considerably more complex than the earlier discussed question of "what to preserve?". In addition to the preserved process, we have to take into consideration the environment we are going to re-deploy the process into. The re-deployment environment will consist of a *fixed* and a *flexible part*. This means that will be an unchangeable (or constrained) part in the re-deployment environment, for example, some machines in a data center, and a changeable (or un-constrained) part of the environment, for example, the possibility of selecting an alternative operating system running on these machines in the data center. We reduce answering the question "what to re-deploy and and why?" to a notion of what is *required* to re-deploy a preserved process. Again, this notion is process-specific, even more than in our previous challenges as the re-deployment environment takes a major role in our reasoning problem.

In reasoning, we have to take three instances of our context model into account, which have to be determined first: a model of the *preserved process*, a model of the *constrained environment*, and a model of the *un-constrained environment*. Afterwards we determine all *feasible re-deployment alternatives* and pick an *optimal one*. This is performed by identifying the *difference* between the preserved process and the constrained environment in more detail. There are four possible outcomes of this evaluation:

None The constrained environment is identical to the environment when the process has been preserved. Therefore, combining their models does not introduce inconsistencies into our ontology, and neither our process, nor the environment have to be adapted to re-deploy.

Overlap The preserved process and the constrained environment *overlap*. This means that their combined model contains overlapping subgraphs which address the same issue, meaning which are not allowed to overlap and therefore introduce inconsistencies into the ontology. For example, two different operating systems on the machines in the data center.

Gap There is a *gap* between the preserved process and the constrained environment. This means that their combined model contain subgraphs which are disconnected from each other although they need to be connected, meaning the disconnected subgraphs introduce inconsistencies into the ontology too. For example, if none of the models cover operating systems.

Both The preserved process and the constrained environment partially *overlap* at one to many points and partially have one to many *gaps* between each other.

After the situation has been sorted out thoroughly, and if we have determined that we cannot immediately re-deploy, we continue in a second reasoning step to determine all feasible re-deployment alternatives. This is based on the models of the preserved process, and both environment models (constrained and un-constrained). The reasoner applies the following strategies in solving any gaps or overlaps:

Overlap In case of an overlap between the models of the preserved process and the constrained environment, the reasoner will take parts out of the model of the preserved process to find options that eliminate the inconsistency from our ontology. This may mean that the reasoner takes larger parts out of the model than the actual overlap, which are filled by parts from the model of the un-constrained environment.

Gap In case of a gap between the models, the reasoner uses the model of the un-constrained environment to find all options to fill this gap and thus eliminate the inconsistency from the model. This may even mean that the reasoner has to take out parts from the model of the preserved process.

Afterwards, all alternatives are ranked to conclude the optimal re-deployment alternative. We are in the process of implementing this reasoning procedure based on satisfyability solvers, specifically the APT-PBO solver [16], which allows us to determine many feasible re-deployment alternatives and rank them according to a process-specific cost function. APT-PBO is different from other similar solvers in that it acts as an interactive system and as such the proposed solutions can be navigated and further decisions taken that is likely to be important in the re-deployment scenario.

An illustrative example of a technical scenario is having a preserved software library (used by a business application) that will not work with the re-deployment environment. The library may have had a known security flaw meaning that in a re-deployment environment it would have to be updated

to a version that included the security fix. Another possible issue could be that the library cannot be used because of licensing issues or doesn't work in combination with some other system that is in place in the new environment. The reasoner would then, based on the context models, try to determine feasible alternatives to the library to update it and rank them according to criteria. This procedure involves the reasoner trying to determine what else would be affected by updating the library. If other software is affected by the update, this could additionally be notified to the "re-deployment manager" and then either a more updated version can be installed or a manually-proposed alternative be applied which fulfils the requirements.

4. CONCLUSION

In this paper, we have motivated the necessity for digital preservation research on dynamic digital objects, such as processes generating (a) dynamic websites, (b) results in e-science experiments, or (c) meta-data. Based on this, we have illustrated three challenges in decision making in the context of the three procedures linked with digital preservation of business processes (planning, execution and redeployment). These challenges were: (1) identifying digital objects a business process depends on; (2) identifying significant changes in digital objects; (3) determining a re-deployment setting.

We have presented a decision support architecture to assist in decision making of these three preservation procedures in general. The architecture has been based on a knowledge representation technique specifically tailored to process preservation, called the *context model*. And we have presented in detail how we are addressing the challenges using the architecture and reasoners that are applicable to our model—in general, logic-based reasoning engines (Pellet and APT-PBO) are being applied. For example, to address the challenge *of how to re-deploy*, we have proposed to integrate existing solvers with the context model. This system will take a set of inputs and try to determine feasible re-deployment solutions based on the preserved system and the target environment.

Furthermore, we have presented the novelty of our modeling approach in the domain of digital preservation. Firstly, we have motivated its goals (a) of being "semantically rich" to allow for decision support which can be process-specifically adapted, and (b) of having the ability to capture "structural and behavioral aspects" (an important aspect to preservation of dynamic digital objects, such as aforementioned processes). Secondly, we have identified the gap of previous modeling approaches in addressing these goals. These are approaches which have been focusing on modeling static objects on semantic levels which we have identified as not being sufficient to fulfill the job of providing generic decision support. And, thirdly, we have presented how this gap is closed by our modeling approach towards achieving our goal of a generic architecture for decision support in digital preservation of business processes.

5. ACKNOWLEDGMENTS

The authors would like to acknowledge the funding by the European Commission under the ICT project "TIMBUS" (Project No. 269940, FP7-ICT-2009-6) within the 7th Framework Programme.

6. REFERENCES

[1] J. Barateiro, D. Draws, M. A. Neumann, and S. Strodl. Digital Preservation Challenges on Software Life Cycle. 2012.

[2] C. Becker and A. Rauber. Decision criteria in digital preservation: What to measure and how. *JASIST*, 62(6), 2011.

[3] A. Calì, T. Lukasiewicz, L. Predoiu, and H. Stuckenschmidt. Journal on Data Semantics. chapter Tightly Coupled Probabilistic Description Logic Programs for the Semantic Web. Springer, 2009.

[4] A. Cheng, J. Esparza, and J. Palsberg. Complexity Results for 1-safe Nets. In *FSTTCS*, 1993.

[5] M. Day. Metadata for Digital Preservation: A Review of Recent Developments. In *ECDL*. Springer, 2001.

[6] DigitalPreservationEurope Partners. DPE Digital Preservation Research Roadmap. Public Deliverable D7.2, DPE, 2007.

[7] M. Fernandez-Lopez, A. Gomez-Perez, and N. Juristo. METHONTOLOGY: from Ontological Art towards Ontological Engineering. In *Proceedings of the AAAI97 Spring Symposium Series on Ontological Engineering*, 1997.

[8] I. Horrocks, P. F. Patel-Schneider, H. Boley, S. Tabet, B. Grosof, and M. Dean. Swrl: A semantic web rule language combining owl and ruleml. W3c member submission, World Wide Web Consortium, 2004.

[9] International Organization For Standardization. OAIS: Open Archival Information System – Reference Model. 2003. ISO 14721:2003.

[10] R. A. Kowalski. Predicate Logic as Programming Language. In *IFIP Congress*, 1974.

[11] R. Mayer, A. Rauber, M. A. Neumann, J. Thomson, and G. Antunes. Preserving Scientific Processes from Design to Publications. In *TPDL*, LNCS. Springer, 2012.

[12] M. A. Neumann, T. Riedel, P. Taylor, H. R. Schmidtke, and M. Beigl. Monitoring for Digital Preservation of Processes. In *CONTEXT*, LNCS. Springer, 2011.

[13] L. Pomello, G. Rozenberg, and C. Simone. A survey of equivalence notions for net based systems. In *Advances in Petri Nets 1992*, LNCS. Springer, 1992.

[14] Premis Editorial Committee. Data Dictionary for Preservation Metadata: PREMIS version 2.0. (March), 2008.

[15] E. Sirin, B. Parsia, B. C. Grau, A. Kalyanpur, and Y. Katz. Pellet: A practical OWL-DL reasoner. *Web Semantics*, 5(2), 2007.

[16] P. Trezentos, I. Lynce, and A. L. Oliveira. Apt-pbo: solving the software dependency problem using pseudo-boolean optimization. In *ASE'10*. ACM, 2010.

[17] M. Uschold and M. Gruninger. Ontologies: Principles, Methods and Applications. *Knowledge Engineering Review*, 11(2), 1996.

[18] J. A. Zachman. A framework for information systems architecture. *IBM systems journal*, 26(3), 1987.