Verification of Semantically-Enhanced Artifact Systems*

Babak Bagheri Hariri, Diego Calvanese, Marco Montali,
Ario Santoso, and Dmitry Solomakhin

KRDB Research Centre for Knowledge and Data, Free University of Bozen-Bolzano
`lastname@inf.unibz.it`

Abstract. Artifact-Centric systems have emerged in the last years as a suitable framework to model business-relevant entities, by combining their static and dynamic aspects. In particular, the Guard-Stage-Milestone (GSM) approach has been recently proposed to model artifacts and their lifecycle in a declarative way. In this paper, we enhance GSM with a Semantic Layer, constituted by a full-fledged OWL 2 QL ontology linked to the artifact information models through mapping specifications. The ontology provides a conceptual view of the domain under study, and allows one to understand the evolution of the artifact system at a higher level of abstraction. In this setting, we present a technique to specify temporal properties expressed over the Semantic Layer, and verify them according to the evolution in the underlying GSM model. This technique has been implemented in a tool that exploits state-of-the-art ontology-based data access technologies to manipulate the temporal properties according to the ontology and the mappings, and that relies on the GSMC model checker for verification.

1 Introduction

In the last decade, the marriage between processes and data has been increasingly advocated as a key objective towards a comprehensive modeling and management of complex enterprises [10]. This requires to go beyond classical (business) process specification languages, which largely leave the connection between the process dimension and the data dimension underspecified, and to consider data and processes as "two sides of the same coin" [21]. In this respect, artifact-centric systems [19,16] have lately emerged as an effective framework to model business-relevant entities, by combining in a holistic way their static and dynamic aspects. Artifacts are characterized by an "information model", which maintains the artifact data, and by a lifecycle that specifies the allowed ways to progress the information model. Among the different proposals for artifact-centric process modelling, the Guard-Stage-Milestone (GSM) approach has been recently proposed to model artifacts and their lifecycle in a declarative, flexible way [17]. GSM is equipped with a formal execution semantics [13], which unambiguously characterizes the artifact progression in response to external events. Several key constructs of the OMG standard on Case Management and Model Notation [1] have been borrowed from GSM.

* This research has been partially supported by the EU under the ICT Collaborative Project ACSI (Artifact-Centric Service Interoperation), grant agreement n. FP7-257593.

[1] `http://www.omg.org/spec/CMMN/`

Despite the tight integration between the data and process component, the artifact information model typically relies on relatively simple structures, such as (nested) lists of key-value pairs. This causes an abstraction gap between the high-level, conceptual view that business stakeholders have of domain-relevant entities and relations, and the low-level representation adopted inside artifacts. To overcome this problem, in [9] it is proposed to enhance artifact systems with a Semantic Layer, constituted by a full-fledged ontology linked to the artifact information models through mapping specifications. On the one hand, the ontology allows one to understand the evolution of the artifact system at a higher level of abstraction. On the other hand, mapping specifications allow one to connect the elements present in the Semantic Layer with the concrete data in the artifact information models, relying on the Ontology-Based Data Access (OBDA) [6].

We follow here an approach similar to [9], and which is based on specifying in terms of the Semantic Layer dynamic/temporal laws that the system should obey, and that need to be verified according to the evolution in the underlying artifact layer. Differently from [9], in which the Semantic Layer is mainly used to *govern* the progression of artifacts, by forbidding the execution of actions that would lead to violation of the constraints in the ontology, here we are primarily interested in exploiting the Semantic Layer to ease the specification of the dynamic/temporal laws. In this light, we extend the technique provided in [9] by relying on a more expressive verification formalism, which supports first-order epistemic queries embedded into an expressive temporal language, the first-order μ-calculus [15], while allowing for quantification across states. The latter makes it possible to predicate over the temporal evolution of individuals, an enhancement that is fundamental for capturing many practical scenarios. To specify the ontology constituting the Semantic Layer, we adopt the OWL 2 QL profile [18] of the standard Web Ontology Language (OWL) [4], since it enjoys so-called *first-order rewritability* of query answering [8], which guarantees that conjunctive queries posed over the ontology can be rewritten into first-order queries that incorporate the ontological constraints, and thus do not require further inference for query answering.

This framework has led to the development of a tool called OBGSM, which relies on GSM as the artifact model, and on state of the art technologies for dealing with the ontology and the mappings, and for performing verification. We refer to an extended version of this paper [3] for proofs and the application of OBGSM on a real case study.

2 Preliminaries

OWL 2 QL is a profile of the Web Ontology Language OWL 2 standardized by the W3C. OWL 2 QL is specifically designed for building an ontology layer to wrap possibly very large data sources. Technically, OWL 2 QL is based on the description logic *DL-Lite*$_{\mathcal{R}}$, which is a member of the *DL-Lite* family [8], designed specifically for effective ontology-based data access, and which we adopt in the following.

In description logics (DLs) [1], the domain of interest is modeled by means of *concepts*, representing classes of objects, and *roles*, representing binary relations between objects.[2] In *DL-Lite*$_{\mathcal{R}}$, concepts C and roles U obey to the following syntax:

[2] Without loss of generality, we do not distinguish between roles (OWL 2 object properties), and attributes (OWL 2 data properties) - see [6]).

$$B ::= N \mid \exists U \qquad C ::= B \mid \exists U.B \qquad U ::= P \mid P^-$$

P denotes a *role name*, and P^- an *inverse role*, which swaps the first and second components of P. N denotes a *concept name*, and B a *basic concept*, which is either simply a concept name, or the projection of a role P on its first component ($\exists P$) or its second component ($\exists P^-$). In the concept $\exists U.B$, the projection on the first (resp., second) component of U can be further qualified by requiring that the second (resp., first) component of U is an instance of the basic concept B.

In DLs, the domain knowledge is split into an intensional part (*TBox*), and an extensional part (*ABox*). Specifically, a *DL-Lite*$_\mathcal{R}$ *ontology* is a pair (T, A), where the TBox T is a finite set of (concept and role) *inclusion assertions* of the forms $B \sqsubseteq C$ and $U_1 \sqsubseteq U_2$, and of *disjointness assertions* of the forms $\mathsf{disjoint}(B_1, B_2)$ and $\mathsf{disjoint}(U_1, U_2)$. The ABox A is a finite set of *facts* (membership assertions) of the forms $N(c_1)$ and $P(c_1, c_2)$, where N and P occur in T, and c_1 and c_2 are constants.

The semantics of a *DL-Lite*$_\mathcal{R}$ ontology is given in terms of first-order *interpretations* $\mathcal{I} = (\Delta^\mathcal{I}, \cdot^\mathcal{I})$, where $\Delta^\mathcal{I}$ is the interpretation domain and $\cdot^\mathcal{I}$ is an interpretation function that assigns to each concept C a subset $C^\mathcal{I} \subseteq \Delta^\mathcal{I}$ and to each role U a binary relation $U^\mathcal{I} \subseteq \Delta^\mathcal{I} \times \Delta^\mathcal{I}$, capturing the intuitive meaning of the various constructs (see [8] for details). An interpretation that satisfies all assertions in T and A is called a *model* of the ontology (T, A), and the ontology is said to be *satisfiable* if it admits at least one model.

Queries. As usual (cf. OWL 2 QL), answers to queries are formed by terms denoting individuals explicitly mentioned in the ABox. The *domain of an ABox A*, denoted by $\textsc{adom}(A)$, is the (finite) set of terms appearing in A. A *union of conjunctive queries* (UCQ) q over a KB (T, A) is a FOL formula of the form $\bigvee_{1 \leq i \leq n} \exists \vec{y_i}.conj_i(\vec{x}, \vec{y_i})$ with free variables $\vec{x}$ and existentially quantified variables $\vec{y}_1, \ldots, \vec{y}_n$. Each $conj_i(\vec{x}, \vec{y_i})$ in q is a conjunction of atoms of the form $N(z)$, $P(z, z')$, where N and P respectively denote a concept and a role name occurring in T, and z, z' are constants in $\textsc{adom}(A)$ or variables in $\vec{x}$ or $\vec{y_i}$, for some $i \in \{1, \ldots, n\}$. The *(certain) answers* to q over (T, A) is the set $ans(q, T, A)$ of substitutions σ of the free variables of q with constants in $\textsc{adom}(A)$ such that $q\sigma$ evaluates to true in every model of (T, A). If q has no free variables, then it is called *boolean* and its certain answers are either true or false.

We compose UCQs using ECQs, i.e., queries of the query language *EQL-Lite*(UCQ) [7], which is the FOL query language whose atoms are UCQs. An *ECQ* over T and A is a possibly open formula of the form

$$Q := [q] \mid \neg Q \mid Q_1 \wedge Q_2 \mid \exists x.Q$$

where q is a UCQ. The *answer to Q over* (T, A) is the set $\textsc{ans}(Q, T, A)$ of tuples of constants in $\textsc{adom}(A)$ defined by composing the certain answers $ans(q, T, A)$ of UCQs q through first-order constructs, and interpreting existentials as ranging over $\textsc{adom}(A)$.

Finally, we recall that *DL-Lite*$_\mathcal{R}$ enjoys the *FO rewritability* property, which states that for every UCQ q, $ans(q, T, A) = ans(\textsc{rew}(q), \emptyset, A)$, where $\textsc{rew}(q)$ is a UCQ computed by the reformulation algorithm in [6]. Notice that this algorithm can be extended to ECQs [7], and that its effect is to "compile away" the TBox. Similarly, ontology satisfiability is FO rewritable for *DL-Lite*$_\mathcal{R}$ TBoxes [6], which states that for every

TBox T, there exists a boolean first-order query $\mathsf{q}_{\mathsf{unsat}}(T)$ such that for every non-empty ABox A, we have that (T, A) is satisfiable iff $ans(\mathsf{q}_{\mathsf{unsat}}(T), T, A) = \mathsf{false}$.

Ontology-Based Data Access (OBDA). In an OBDA system, a relational database is connected to an ontology representing the domain of interest by a mapping, which relates database values with values and (abstract) objects in the ontology (cf. [6]). In particular, we make use of a countably infinite set $\mathcal{V}$ of values and a set Λ of function symbols, each with an associated arity. We also define the set $\mathcal{C}$ of constants as the union of $\mathcal{V}$ and the set $\{f(d_1, \ldots, d_n) \mid f \in \Lambda$ and $d_1, \ldots, d_n \in \mathcal{V}\}$ of *object terms*.

Formally, an OBDA system is a structure $\mathcal{O} = \langle R, T, \mathcal{M} \rangle$, where: *(i)* $R = \{R_1, \ldots, R_n\}$ is a database schema, constituted by a finite set of relation schemas; *(ii)* T is a *DL-Lite*$_{\mathcal{R}}$ TBox; *(iii)* $\mathcal{M}$ is a set of mapping assertions, each of the form $\Phi(\vec{x}) \rightsquigarrow \Psi(\vec{y}, \vec{t})$, where: (a) $\vec{x}$ is a non-empty set of variables, (b) $\vec{y} \subseteq \vec{x}$, (c) $\vec{t}$ is a set of object terms of the form $f(\vec{z})$, with $f \in \Lambda$ and $\vec{z} \subseteq \vec{x}$, (d) $\Phi(\vec{x})$, which also called as *source query*, is an arbitrary SQL query over R, with $\vec{x}$ as output variables, and (e) $\Psi(\vec{y}, \vec{t})$, which also called as *target query*, is a CQ over T of arity $n > 0$ without non-distinguished variables, whose atoms are over the variables $\vec{y}$ and the object terms $\vec{t}$.

Given a database instance $\mathcal{I}$ (made up of values in $\mathcal{V}$ and conforming to schema R) and a mapping assertion $m = \Phi(x) \rightsquigarrow \Psi(y, t)$, the *virtual ABox* generated from $\mathcal{I}$ by m is $m(\mathcal{I}) = \bigcup_{v \in eval(\Phi, \mathcal{I})} \Psi[x/v]$, where $eval(\Phi, \mathcal{I})$ denotes the evaluation of the SQL query Φ over $\mathcal{I}$, and where we consider $\Psi[x/v]$ to be a set of atoms (as opposed to a conjunction). The ABox generated from $\mathcal{I}$ by the mapping $\mathcal{M}$ is $\mathcal{M}(\mathcal{I}) = \bigcup_{m \in \mathcal{M}} m(\mathcal{I})$. As for ABoxes, the active domain $\text{ADOM}(\mathcal{I})$ of a database instance $\mathcal{I}$ is the set of values occurring in $\mathcal{I}$. Given an OBDA system $\mathcal{O} = \langle R, T, \mathcal{M} \rangle$ and a database instance $\mathcal{I}$ for R, a *model* for $\mathcal{O}$ wrt $\mathcal{I}$ is a model of the ontology $(T, \mathcal{M}(\mathcal{I}))$. We say that $\mathcal{O}$ wrt $\mathcal{I}$ is satisfiable if it admits a model wrt $\mathcal{I}$.

A UCQ q over an OBDA system $\mathcal{O} = \langle R, T, \mathcal{M} \rangle$ and a relational instance $\mathcal{I}$ for R is simply an UCQ over $(T, \mathcal{M}(\mathcal{I}))$. To compute the certain answers of q over $\mathcal{O}$ wrt $\mathcal{I}$, we follow the standard three-step approach [6]: *(i)* q is *rewritten* to compile away T, obtaining $q_r = rew(q, T)$; *(ii)* the mapping $\mathcal{M}$ is used to *unfold* q_r into an SQL query over R, denoted by $\text{UNFOLD}(q_r, \mathcal{M})$ [20]; *(iii)* such a query is executed over $\mathcal{I}$, obtaining the certain answers. For an ECQ, we can proceed in a similar way, applying the rewriting and unfolding steps to the embedded UCQs. It follows that computing certain answers to UCQs/ECQs in an OBDA system is FO rewritable. Applying the unfolding step to $\mathsf{q}_{\mathsf{unsat}}(T)$, we obtain also that satisfiability in $\mathcal{O}$ is FO rewritable.

3 Semantically-Enhanced Artifact Systems

In this section we introduce Semantically-enhanced Artifact Systems (SASs), taking inspiration from the semantic governance framework introduced in [9]. Intuitively, SAS models systems in which artifacts progress according to their lifecycles, and in which the evolution of the entire system is understood through the conceptual lens of an OWL 2 QL ontology. In accordance with the literature [14], it is assumed that artifacts are equipped with a relational information model. More specifically, a SAS is constituted by: *(i)* A *Relational Layer*, which account for the (relational) information models of the artifacts, and which employs a global transition relation to abstractly capture the

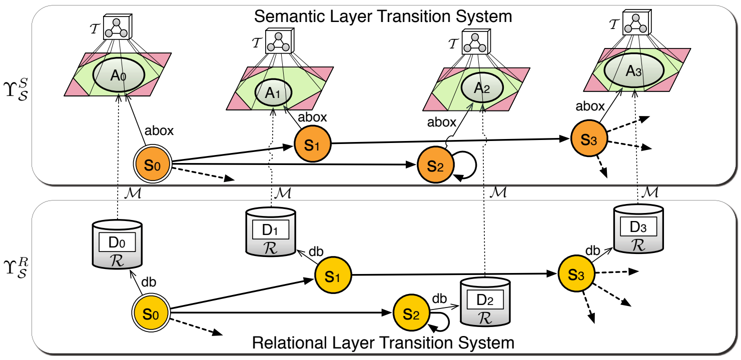
Semantic Layer Transition System
Relational Layer Transition System
$\Upsilon_{\mathcal{S}}^{S}$
$\Upsilon_{\mathcal{S}}^{R}$
$\mathcal{T}$
A0
A1
A2
A3
abox
S0
S1
S2
S3
$\mathcal{M}$
D0
D1
D2
D3
$\mathcal{R}$
db

Fig. 1. Sketch of the Relational and Semantic Transition System of a SAS

step-by-step evolution of the system as a whole. *(ii)* A *Semantic Layer*, which contains an OWL 2 QL ontology that conceptually accounts for the domain under study. *(iii)* A set of *mapping assertions* describing how to virtually project data concretely maintained at the Relational Layer into concepts and relations modeled in the Semantic Layer, thus providing a link between the artifact information models and the ontology.

In the following, we assume a countably infinite set of values $\mathcal{V}$. Formally, a SAS $\mathcal{S}$ is a tuple $\mathcal{S} = \langle R, \mathcal{I}_0, \mathcal{F}, T, \mathcal{M} \rangle$, where: *(i)* R is a database schema that incorporates the schemas of all artifact information models present in the Relational Layer; *(ii)* $\mathcal{I}_0$ is a database instance made up of values in $\mathcal{V}$ and conforming to R, which represents the initial state of the Relational Layer,; *(iii)* $\mathcal{F} \subseteq \Gamma \times \Gamma$ is the transition relation that describes the overall progression mechanism of the Relational Layer, where Γ is the set of all instances made up of values in $\mathcal{V}$ and conforming to R; *(iv)* T is a *DL-Lite*$_{\mathcal{R}}$ TBox; *(v)* $\mathcal{M}$ is a set of mapping assertions that connect R to T, following the approach described in Section 2. The triple $\langle R, T, \mathcal{M} \rangle$ constitutes, in fact, an OBDA system. Thus, $\mathcal{S}$ can be seen as an OBDA system equipped with a transition relation that accounts for the dynamics of the system at the level of R, starting from $\mathcal{I}_0$.

3.1 Execution Semantics

The execution semantics of a SAS $\mathcal{S}$ is provided by means of transition systems. While the temporal structure of such systems is fully determined by the transition relation of $\mathcal{S}$, the content of each state in the system depends on whether the dynamics of SASs is understood directly at the Relational Layer, or through the conceptual lens of the Semantic Layer ontology. In the former case, each state is associated to a database instance that represents the current snapshot of the artifact information models, whereas in the latter case each state is associated to an ABox that represents the current state of the system, as understood by the Semantic Layer.

Following this approach, the execution semantics of $\mathcal{S}$ is captured in terms of two transition systems, one describing the allowed evolutions at the Relational Layer (*Relational Transition System - RTS*), and one abstracting them at the Semantic Layer (*Semantic Transition System* - STS). Figure 1 provides a graphical intuition about the RTS and STS, and their interrelations.

RTS. Given a SAS $\mathcal{S} = \langle R, \mathcal{I}_0, \mathcal{F}, T, \mathcal{M} \rangle$, its RTS $\Upsilon_{\mathcal{S}}^{\mathrm{R}}$ is defined as a tuple $\langle R, \Sigma, s_0, db, \Rightarrow \rangle$, where: *(i)* Σ is a set of states, *(ii)* $s_0 \in \Sigma$, *(iii)* db is a function that, given a state in Σ, returns a corresponding database instance (conforming to R), *(iv)* $\Rightarrow \subseteq \Sigma \times \Sigma$ is the transition relation. The components Σ, $\Rightarrow$ and db of $\Upsilon_{\mathcal{S}}^{\mathrm{R}}$ are defined by simultaneous induction as the smallest sets satisfying the following conditions:

- $db(s_0) = \mathcal{I}_0$;
- for every databases instance $\mathcal{I}'$ such that $\langle db(s), \mathcal{I}' \rangle \in \mathcal{F}$:
  - if there exists $s' \in \Sigma$ such that $db(s') = \mathcal{I}'$, then $s \Rightarrow s'$;
  - otherwise, if $\mathcal{O} = \langle R, T, \mathcal{M} \rangle$ is *satisfiable* wrt $\mathcal{I}'$, then $s' \in \Sigma$, $s \Rightarrow s'$ and $db(s') = \mathcal{I}'$, where s' is a fresh state.

The satisfiability check done in the last step of the RTS construction accounts for the semantic *governance* (cf. Section 1): a transition is preserved in the RTS only if the target state does not violate any constraints of the Semantic Layer, otherwise it is rejected [9].

STS. Given a SAS $\mathcal{S} = \langle R, \mathcal{I}_0, \mathcal{F}, T, \mathcal{M} \rangle$, its STS $\Upsilon_{\mathcal{S}}^{\mathrm{S}}$ is defined as a tuple $\langle R, \Sigma, s_0, db, \Rightarrow \rangle$, which is similar to an RTS, except from the fact that states are attached to ABoxes, not database instances. In particular, $\Upsilon_{\mathcal{S}}^{\mathrm{S}}$ is defined as a "virtualization" of the RTS $\Upsilon_{\mathcal{S}}^{\mathrm{R}} = \langle R, \Sigma, s_0, db, \Rightarrow \rangle$ at the Semantic Layer: it maintains the structure of $\Upsilon_{\mathcal{S}}^{\mathrm{R}}$ unaltered, reflecting that the progression of the system is determined at the Relational Layer, but it associates each state to a virtual ABox obtained from the application of the mapping specification $\mathcal{M}$ to the database instance associated by $\Upsilon_{\mathcal{S}}^{\mathrm{R}}$ to the same state. Formally, the transition relation $\mathcal{M}$ is equivalent to the one of the $\mathcal{S}$, and the $abox$ function of $\Upsilon_{\mathcal{S}}^{\mathrm{S}}$ is defined as follows: for each $s \in \Sigma$, $abox(s) = \mathcal{M}(db(s))$.

4 Verification of Semantically-Enhanced Artifact Systems

Given a SAS $\mathcal{S}$, we are interested in studying verification of semantic dynamic/temporal properties specified over the Semantic Layer, i.e., to be checked against the STS $\Upsilon_{\mathcal{S}}^{\mathrm{S}}$. As verification formalism, we consider a variant of first-order μ-calculus [15,22], called $\mu\mathcal{L}_A^{\mathrm{EQL}}$ [2,11]. We observe that μ-calculus is one of the most powerful temporal logics: it subsumes LTL, PSL, and CTL* [12]. The logic $\mu\mathcal{L}_A^{\mathrm{EQL}}$ supports querying the states of the STS through the first-order epistemic queries introduced in Section 2. In $\mu\mathcal{L}_A^{\mathrm{EQL}}$, first-order quantification is restricted to objects present in the current ABox, and can be used to relate objects across states. The syntax of $\mu\mathcal{L}_A^{\mathrm{EQL}}$ is as follows:

$$\Phi ::= Q \mid \neg\Phi \mid \Phi_1 \wedge \Phi_2 \mid \exists x.\textsc{live}(x) \wedge \Phi \mid \langle - \rangle \Phi \mid Z \mid \mu Z.\Phi$$

Where Q is an ECQ over T, Z is a second-order variable denoting a 0-ary predicate, μ is the least fixpoint operator, and the special predicate $\textsc{live}(x)$ is used to indicate that x belongs to the current active domain, i.e., it is mentioned in some concept or role of the current ABox. For a detailed semantics of $\mu\mathcal{L}_A^{\mathrm{EQL}}$, refer to [2,11].

Given a SAS $\mathcal{S}$, we show that verification of $\mu\mathcal{L}_A^{\mathrm{EQL}}$ properties over the STS $\Upsilon_{\mathcal{S}}^{\mathrm{S}}$ can be reduced to verification of $\mu\mathcal{L}_A$ [10] properties over the RTS $\Upsilon_{\mathcal{S}}^{\mathrm{R}}$, where $\mu\mathcal{L}_A$ is a logic similar to $\mu\mathcal{L}_A^{\mathrm{EQL}}$, except for the local formula Q, which is an (open) first-order query over the database schema in the Relational Layer.

Theorem 1. *For every SAS $\mathcal{S}$ and $\mu\mathcal{L}_A^{\mathrm{EQL}}$ property Φ, there exists a $\mu\mathcal{L}_A$ property Φ' such that $\Upsilon_{\mathcal{S}}^{\mathrm{R}}$ satisfies Φ if and only if $\Upsilon_{\mathcal{S}}^{\mathrm{S}}$ satisfied Φ'.*

5 SAS Instantiation: The OBGSM Tool

The formal framework of SASs has led to the development of the OBGSM tool. OBGSM assumes that the RTS is obtained from artifacts, specified using the Guard-Stage-Milestone (GSM) approach. The main task accomplished by the tool is the reformulation of temporal properties expressed over the ontology in terms of the underlying GSM information model. In particular, OBGSM adopts: *(i)* The state of the art OBDA system -ONTOP-[3] to efficiently rewrite and unfold the epistemic queries embedded in the temporal property to verify. *(ii)* the recently developed GSMC model checker for GSM [5] to accomplish the actual verification phase. GSMC is currently the only model checker able to verify temporal formulae over artifact systems.

Since the temporal formalism supported by GSMC is a variant of the first-order branching time logic CTL [12] with a restricted form of quantification across states, the $\mu\mathcal{L}_A^{\mathrm{EQL}}$ has been restricted accordingly in the tool [4]. This, in turn, required to suitably accommodate the mapping language so as to ensure that temporal formulae over the Semantic Layer correspond, once rewritten and unfolded, to properties that can be processed by GSMC. Furthermore, in accordance to -ONTOP-, both the temporal properties and the mappings rely on the SPARQL query language to query the Semantic Layer.

For a more comprehensive description of the tool, and its application to a real-world case study in the energy domain, developed within the ACSI Project, please refer to [3].

6 Discussion

The OBGSM tool works under the assumption that the Semantic Layer is used to enhance GSM, but not to govern it. In fact, the construction of the RTS is handled internally by GSMC, it is not possible (at least for the time being) to prune it so as to remove inconsistent states. Hence, OBGSM must assume that all the states in the RTS are consistent with the constraints of the Semantic Layer. This can be trivially achieved by, e.g., avoiding to use negative inclusion assertions in the TBox, which are the only source of inconsistency for OWL 2 QL. If inconsistent states can be generated by the GSM specification, the strategy of delegating the verification to GSMC as a black box cannot be followed directly. One possible solution to this is to minimally change the GSMC implementation by introducing a test to detect states that should not be added to the RTS during the construction, then implementing it as a satisfiability check wrt the ontology. The other possible solution is to consider fragments of the verification logic, and investigate whether the check for consistency can be embedded in the formula to verify, so as to avoid any impact on GSMC. These scenarios provide us with interesting problems for future investigation.

References

1. Baader, F., Calvanese, D., McGuinness, D., Nardi, D., Patel-Schneider, P.F. (eds.): The Description Logic Handbook: Theory, Implementation and Applications. Cambridge University Press (2003)
2. Bagheri Hariri, B., Calvanese, D., Montali, M., De Giacomo, G., De Masellis, R., Felli, P.: Description logic Knowledge and Action Bases. J. of Artificial Intelligence Research (2013)

[3] `http://ontop.inf.unibz.it/`

[4] Notice that CTL can be expressed in the alternation-free fragment of the μ-calculus [15].

3. Bagheri Hariri, B., Calvanese, D., Montali, M., Santoso, A., Solomakhin, D.: Verification of semantically-enhanced artifact systems (extended version). CoRR (2013)
4. Bao, J., et al.: OWL 2 Web Ontology Language document overview, 2nd edn. W3C Recommendation. World Wide Web Consortium (2012)
5. Belardinelli, F., Lomuscio, A., Patrizi, F.: Verification of GSM-based artifact-centric systems through finite abstraction. In: Liu, C., Ludwig, H., Toumani, F., Yu, Q. (eds.) ICSOC 2012. LNCS, vol. 7636, pp. 17–31. Springer, Heidelberg (2012)
6. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Poggi, A., Rodriguez-Muro, M., Rosati, R.: Ontologies and databases: The *DL-Lite* approach. In: Tessaris, S., Franconi, E., Eiter, T., Gutierrez, C., Handschuh, S., Rousset, M.-C., Schmidt, R.A. (eds.) Reasoning Web 2009. LNCS, vol. 5689, pp. 255–356. Springer, Heidelberg (2009)
7. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: EQL-Lite: Effective first-order query processing in description logics. In: Proc. of IJCAI 2007, pp. 274–279 (2007)
8. Calvanese, D., De Giacomo, G., Lembo, D., Lenzerini, M., Rosati, R.: Tractable reasoning and efficient query answering in description logics: The *DL-Lite* family. J. of Automated Reasoning 39(3), 385–429 (2007)
9. Calvanese, D., De Giacomo, G., Lembo, D., Montali, M., Santoso, A.: Ontology-based governance of data-aware processes. In: Krötzsch, M., Straccia, U. (eds.) RR 2012. LNCS, vol. 7497, pp. 25–41. Springer, Heidelberg (2012)
10. Calvanese, D., De Giacomo, G., Montali, M.: Foundations of data aware process analysis: A database theory perspective. In: Proc. of PODS 2013 (2013)
11. Calvanese, D., Kharlamov, E., Montali, M., Santoso, A., Zheleznyakov, D.: Verification of inconsistency-tolerant knowledge and action bases. In: Proc. of IJCAI 2013 (2013)
12. Clarke, E.M., Grumberg, O., Peled, D.A.: Model checking. The MIT Press (1999)
13. Damaggio, E., Hull, R., Vaculín, R.: On the equivalence of incremental and fixpoint semantics for business artifacts with Guard-Stage-Milestone lifecycles. Information Systems (2013)
14. Deutsch, A., Hull, R., Patrizi, F., Vianu, V.: Automatic verification of data-centric business processes. In: Proc. of ICDT 2009, pp. 252–267 (2009)
15. Emerson, E.A.: Model checking and the Mu-calculus. In: Immerman, N., Kolaitis, P. (eds.) Proceedings of the DIMACS Symposium on Descriptive Complexity and Finite Models. DIMACS Series in Discrete Mathematics and Theoretical Computer Science, pp. 185–214. American Mathematical Society Press (1996) ISBN 0-8218-0517-7
16. Hull, R.: Artifact-centric business process models: Brief survey of research results and challenges. In: Meersman, R., Tari, Z. (eds.) OTM 2008, Part II. LNCS, vol. 5332, pp. 1152–1163. Springer, Heidelberg (2008)
17. Hull, R., Damaggio, E., De Masellis, R., Fournier, F., Gupta, M., Heath III, F.T., Hobson, S., Linehan, M., Maradugu, S., Nigam, A., Sukaviriya, P.N., Vaculin, R.: Business artifacts with Guard-Stage-Milestone lifecycles: Managing artifact interactions with conditions and events. In: Proc. of the 5th ACM Int. Conf. on Distributed Event-Based Systems, DEBS 2011 (2011)
18. Motik, B., Cuenca Grau, B., Horrocks, I., Wu, Z., Fokoue, A., Lutz, C.: OWL 2 Web Ontology Language profiles, 2nd edn. Tech. rep., W3C Recommendation (2012)
19. Nigam, A., Caswell, N.S.: Business artifacts: An approach to operational specification. IBM Systems J. 42(3), 428–445 (2003)
20. Poggi, A., Lembo, D., Calvanese, D., De Giacomo, G., Lenzerini, M., Rosati, R.: Linking data to ontologies. In: Spaccapietra, S. (ed.) Journal on Data Semantics X. LNCS, vol. 4900, pp. 133–173. Springer, Heidelberg (2008)
21. Reichert, M.: Process and data: Two sides of the same coin? In: Meersman, R., et al. (eds.) OTM 2012, Part I. LNCS, vol. 7565, pp. 2–19. Springer, Heidelberg (2012)
22. Stirling, C.: Modal and Temporal Properties of Processes. Springer (2001)