Artifact-centric Business Process Models in UML: Specification and Reasoning (Extended Abstract)

Montserrat Estañol
supervised by Prof. Ernest Teniente

Universitat Politècnica de Catalunya, Barcelona, Spain
{estanyol|teniente}@essi.upc.edu

1 Introduction

Processes can be modeled from different perspectives. Traditional process modeling has followed the process-centric perspective, where the focus is on the sequencing of activities (i.e. the control flow), largely ignoring the data required by them. In contrast, the artifact-centric (or data-centric) process modeling approach focuses on defining the data required by the tasks or activities, and details of the tasks themselves in terms of the changes they make to the data.

One of the challenges in this context is finding a way to represent business processes from an artifact-centric perspective, in a way that is easy to understand by the people involved in the business process and that is formal at the same time. To achieve this we propose a framework, BAUML, which is based on using a combination of UML and OCL models.

Once a process model has been defined, it is important to ensure its quality. This will avoid the propagation of errors to the process's implementation. Although there are many different quality criteria, we focus on the semantic correctness of the model, answering questions such as *does it represent reality correctly?* or *are there any errors and contradictions in it?*

Therefore, the second part of this thesis is concerned with finding a way to determine the semantic correctness of our BAUML models. To begin with, we propose a translation of a BAUML model into a DCDS (an external framework to our work), to which model checking techniques can be applied to determine the model's correctness. However, DCDSs have been defined theoretically and there is no tool that implements them.

To solve this, we created a prototype tool, AuRUS-BAUML, which is able to translate our BAUML models into first-order logic and to reason on their semantic correctness using an existing tool, SVTe. Another contribution is the logic translation which is performed by the tool. Last but not least, we contribute a study on the decidability of reasoning on the BAUML models.

To the best of our knowledge, this is the first approach using a combination of UML and OCL models for modeling artifact-centric business processes which also provides a way of checking their semantic correctness.

2 Modeling Artifact-centric BPM in UML/OCL

The BALSA framework [3] defines four different dimensions that should be present in any artifact-centric business process model. They are the following: *business artifacts*, which represent meaningful data for the business, and have an ID; *lifecycles*, which represent the evolution of an artifact; *services*, representing tasks (i.e. meaningful units of work) that evolve the artifact; and *associations*, representing constraints in the manner how services make changes to artifacts.

In traditional, process-centric models, such as a flowchart, there is usually only one of the BALSA dimensions represented: the flow between the tasks in the process, called associations in the framework. In terms of artifact-centric approaches, there are several alternatives (see [2] for details). However, none of the examined works has the following characteristics: 1. It uses a well-known, formal language, for all the dimensions in BALSA, which can be understood by business modelers and developers; 2. The language bridges the gap between artifact and process-centric approaches; 3. There are tools to model the business process following the approach.

The modeling approach that we propose to deal with these issues is to use a combination of UML and OCL models. UML class diagrams for business artifacts; UML state machine diagrams for lifecycles; UML activity diagrams for associations, and OCL operation contracts for services or tasks. We call our approach BAUML (BALSA UML, for short).

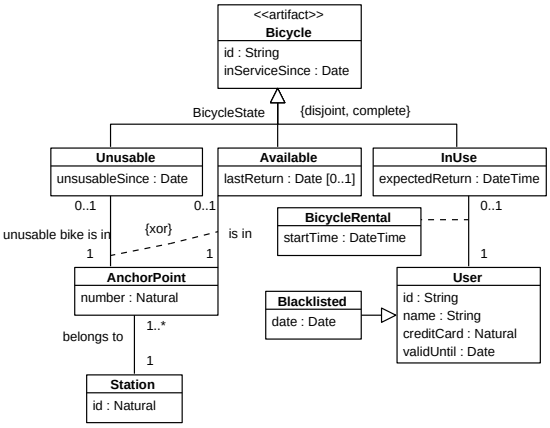

Fig. 1. Class diagram for the *Bicing* example, representing the business artifacts

We will illustrate the approach by means of an example based on a city bicycle rental system (*Bicing*). Business artifacts correspond to some of the classes in the UML class diagram (see Figure 1); they are those whose evolution results in relevant states from the business's point of view (*Bicycle* in our example). We refer to the rest of classes as objects (e.g. *User* or *AnchorPoint*). Due to space

limitations we do not include any integrity constraints, which should be defined in OCL.

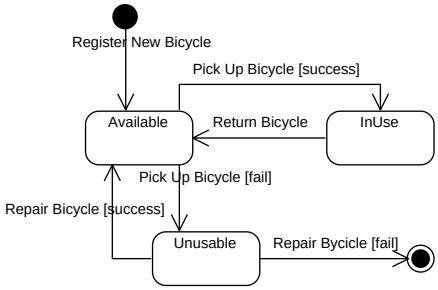

Fig. 2. *Lifecycle* of artifact *Bicycle*.

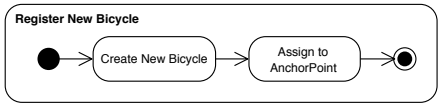

Fig. 3. Act. diagr. of *RegisterNewBicycle*.

For each artifact, a state machine diagram will show its lifecycle (see Figure 2). Each of the states corresponds to one of the subclasses of *Bicycle*, and each transition shows how a *Bicycle* evolves from one state to the next. Then, each transition of the state machine diagram is further specified (as they are not atomic) by means of an activity diagram determining the associations between the services or tasks of the artifact. See the activity diagram of *Register New Bicycle* in Figure 3. Finally, the behavior of the tasks from each activity diagram is defined through an OCL operation contract, which contains a precondition, stating the conditions that must be true in order for the operation to execute, and a postcondition, indicating the state of the system after the execution of the operation. Below we show the operation contract for task *CreateNewBicycle*, which has no precondition and which creates a new bicycle.

```
operation createNewBicycle(bId: String): Bicycle
post: Available.allInstances()->exists (b | b.oclIsNew() and b.uid = bId and
    b.inServiceSince = today() and result = b))
```

The framework presented here has several advantages: it uses two languages, UML and OCL, both of which are ISO standards. They allow us to define the business process at a high level of abstraction and thus they are independent of the final technological implementation. These languages, but more specially UML, can be understood by business modelers and developers. In addition, our proposal has precise semantics and can deal with business processes that contain more than one artifact [2].

3 Reasoning on Artifact-centric Business Process Models

Given a BAUML model, our goal is to be able to ensure that it fulfills a certain set of desirable properties. In particular, we wish to check that there are no errors in the model and that it represents reality correctly. Unfortunately there are no available methods to reason with models used in BAUML.

Data-centric Dynamic Systems (DCDSs) are an alternative way of representing data-centric business process models. They provide a formal representation at a lower level of abstraction than UML and OCL, but it is possible to apply model checking techniques to them in order to check if they fulfill a certain property [1].

To take advantage of DCDSs, the thesis presents and formalizes (by means of algorithms) a translation process to obtain a DCDS that is equivalent to a BAUML model. A relational DCDS is a tuple $\mathcal{S} = \langle \mathcal{D}, \mathcal{P} \rangle$, where $\mathcal{D}$ corresponds to the data layer and $\mathcal{P}$ to the process layer. The data layer contains, among others, a database schema $\mathcal{R}$ and an initial database instance $\mathcal{I}_0$.The process layer contains, among others, a set of actions $\mathcal{A}$ and a set of condition-action rules of the form $\mathcal{Q} \mapsto \alpha$, where $\mathcal{Q}$ is a first-order query and $\alpha \in \mathcal{A}$.

Intuitively, we could assume that one way of performing the translation process from BAUML to DCDSs is by mapping the models according to the BALSA dimension which they represent, e.g. by translating the class diagram into a database schema $\mathcal{R}$ as both represent the business artifacts. However, the translation process requires additional considerations, such as adding tables to track the evolution through the tasks or services in the activity diagram [2].

Once the model has been translated and the property has been defined, we are able to apply model checking techniques to determine if the original BAUML model fulfills said property. To do so, we will also need to provide an initial instance of the database schema.

Unfortunately, there is no tool that can reason with DCDSs to check their semantic correctness. For this reason, we have created a prototype tool, AuRUS-BAUML, which is able to perform several semantic tests to check the correctness of a BAUML model, by translating the starting models into first-order logic. The translation process performed by the tool is a contribution of the thesis based on the work of [4]. Note that this approach does not need an initial instance of the model to obtain results, unlike DCDSs.

In particular, the tool can perform two types of semantic tests: verification and validation. Verification ensures that there are no internal errors and contradictions in the model (e.g. guaranteeing executability of the tasks), and can be generated automatically; validation focuses on external correctness: ensuring that the model fulfills the user requirements (e.g. checking that blacklisted users cannot rent bicycles).

Another contribution of the thesis is a study on the decidability of reasoning for the BAUML framework proposed in it. Determining whether an unrestricted BAUML model fulfills a certain property is undecidable. By establishing certain conditions over them, it is possible to ensure decidability over them without bounding the number of active artifact instances. On the other hand, if the two artifacts share read-write objects, we need to bound the number of active artifact instances to ensure decidability. See [2] for details.

4 Conclusions

The thesis presents a way to model business processes following an artifact-centric approach using a combination of UML and OCL models, which results in a high-level and mainly graphical representation of the process. We call this approach the BAUML framework.

Since the models represent both the relevant data and how the tasks in the processes make changes to this data, we can check the semantic correctness of the models. This is a another contribution of the thesis. We propose two different ways to do. The first one relies on translating the BAUML to a DCDS, an external framework to our own work.

However, as DCDSs have been proposed theoretically, we then prove the feasibility of our approach by implementing a prototype tool, AuRUS-BAUML, which can translate the initial models into first-order logic automatically and generate tests from them. It is also possible to execute user-defined tests. This allows the detection of internal errors and contradictions (verification) and that the model fulfills the requirements (validation).

Finally, a last contribution is a study on the decidability of reasoning on our BAUML models. Determining whether a BAUML model fulfills a certain property is undecidable, but we establish certain conditions to guarantee decidability.

As further work, we would like to incorporate other notations such as BPMN or ER diagrams to make the framework more flexible. Another point to focus on would be to address the limitations of the translation process or the tools (e.g. by making it more efficient).

Acknowledgments: The work was partly supported by Universitat Politècnica de Catalunya, Ministerio de Ciencia e Innovación under project TIN2011-24747, Ministerio de Economía y Competitividad under project TIN2014-52938-C2-2-R and AGAUR agency under project 2014 SGR 1534.

References

1. Bagheri Hariri, B., Calvanese, D., Giacomo, G.D., Deutsch, A., Montali, M.: Verification of relational data-centric dynamic systems with external services. In: Hull, R., Fan, W. (eds.) PODS. pp. 163–174. ACM (2013)
2. Estañol, M.: Artifact-centric Business Process Models in UML: Specification and Reasoning. Ph.D. thesis, Universitat Politècnica de Catalunya (2016), advised by Prof. Ernest Teniente. Available at: http://hdl.handle.net/2117/96329
3. Hull, R.: Artifact-centric business process models: Brief survey of research results and challenges. In: Meersman, R., Tari, Z. (eds.) OTM 2008. LNCS, vol. 5332, pp. 1152–1163. Springer (2008)
4. Queralt, A., Teniente, E.: Reasoning on UML conceptual schemas with operations. In: van Eck, P., Gordijn, J., Wieringa, R. (eds.) CAiSE 2009. pp. 47–62. LNCS, Springer (2009)