Linking Data and BPMN Processes to Achieve Executable Models (Summary) ⋆

Giuseppe De Giacomo[1], Xavier Oriol[2], Montserrat Estañol[2], Ernest Teniente[2]

[1] Sapienza Università di Roma, Rome, Italy
degiacomo@dis.uniroma1.it
[2] Universitat Politècnica de Catalunya, Barcelona, Spain
{oriol,estanyol,teniente}@essi.upc.edu

Summary of the Contribution

The two main assets of any organization are (i) information, i.e., data, which are the things that the organization knows about, and (ii) processes, which are collections of activities that describe how work is performed within an organization. Recent research has highlighted the importance of considering both process and data as key elements in process and service design. However, models that represent processes and data are typically developed by different teams, resulting in unrelated models which neglect the interaction between the two.

Bearing this in mind, in this paper we propose to connect processes and data, and a way to automatically execute the resulting model. To do so, we assume that processes are represented using BPMN and data in a UML class diagram. In order to link both formalisms, we propose the following: (1) the creation of an *Artifact*, which represents a set of process variables associated to a certain process instance, and (2) the specification of the activities or tasks in the process, showing how they make changes to the data. We propose representing the artifact as an additional class in the UML class diagram. On the other hand, we opt for OCL operation contracts (with a pre and a postcondition) to specify details of the process activities. Note that other languages could be used to represent the data, the process and the contracts, as long as they have unambiguous semantics and whose expressiveness is equivalent to first-order logic.

Following this framework, we can then achieve executability of the framework, by relying on SQL technology. The UML class diagram is encoded as a relational database, the BPMN diagram can be formalized as a Petri net, and the OCL contracts can be encoded as logic rules from which SQL statements can be derived and applied to the database.

To prove the feasiblity of our approach, we have developed a prototype tool in Java which can load the models in our framework and execute the operations at runtime in a relational database.