Data-awareness in B2B process modeling

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 181 (2021) 15–22

Procedia Computer Science

www.elsevier.com/locate/procedia

CENTERIS - International Conference on ENTERprise Information Systems / ProjMAN - International Conference on Project MANagement / HCist - International Conference on Health and Social Care Information Systems and Technologies 2020

Data-awareness in B2B process modeling

Giorgio Bruno*

Politecnico di Torino, Corso Duca degli Abruzzi 24, 10129 Torino, Italy

Abstract

This paper presents a notation to model business processes in the context of B2B (Business to Business) systems. Data-awareness is a key asset of the notation: data is placed at the forefront of process representations and task execution is data-driven instead of control-driven. In addition to process models, the notation (B2BPN) includes three other types of models, i.e. the architectural model, the collaboration models and the information model of the process. Data can transit between B2B processes through the interactions defined in the collaboration models and such models represent the relationships between the companies taking part in B2B systems. The notation is illustrated with an example concerning the order management of a distributor.

Keywords: business processes; BPM notations; dataflow; data models; collaboration models

1. Introduction

The notion of data-awareness in business process models can take on various meanings. The simplest is the graphical representation of the data handled by processes: the BPMN (Business Process Model & Notation) standard

* Corresponding author.
E-mail address: giorgio.bruno@polito.it

10.1016/j.procs.2021.01.094

[1] offers data objects to represent the input and output data of tasks but without precisely defining a data-driven behavior of the tasks. Moreover, data objects are representations of the variables of process instances.

The notion of process instance is suitable for handling a single data and any other complementary data: the underlying assumption is that there is no interference between instances.

In many cases, this assumption does not hold and this has led to the identification of new models in which a singleton process (i.e. without instances) can handle a dataflow subject to modifications, decompositions and aggregations. Enterprise Integration Patterns [2] provide a detailed analysis of the ways a dataflow can be transformed.

In such models, the notions of dataflow and tasks have equal importance and the execution of tasks is data-driven. The data in a singleton process are no longer representations of variables but are actual data whose existence is separate from the process even if data is influenced by the process. The data can reside in databases or in message queues as proposed in the Enterprise Integration Patterns.

This paper focuses on singleton processes which operate in a B2B (Business-to-Business) scenario where various companies (e.g., clients, suppliers and distributors) interact through collaboration protocols. Such processes are named B2B processes: a B2B process belongs to a company and its purpose is to enable the company to handle collaborations with partners.

The notation, named B2BPN (B2B Process Notation), presented in this paper provides four models. The architectural model indicates the collaborations between the company under consideration and its partners. The collaboration models define the details in terms of interactions. The B2B process model shows how the company handles the collaborations with its partners. The information model addresses the (data) entities the B2B process acts on.

An example concerning the order management of a distributor is provided. The distributor receives purchase orders from clients and sends purchase orders to suppliers. The lines of a client order can be aggregated with the lines of other client orders in various supplier orders: the relationship between client orders and supplier orders is many to many, and therefore the order handling process is a singleton one because a client order cannot be handled with a single instance separate from the others. With BPMN, it is difficult to handle situations like this [3].

This paper is organized as follows. Section 2 is about the related work, section 3 illustrates the features of B2BPN with the help of the order management example, and section 4 contains the conclusion.

2. Related work

Current notations for business process modeling can be divided into several categories based on the aspects they focus on.

An activity-centric notation such as the BPMN standard is instance-oriented and places emphasis on the control flow, which determines the precedence between tasks based on completion events. BPMN can also represent data but in a way that is judged as an “afterthought” [4]. The control flow and the dataflow are separate and this makes the notation complicated.

An extension [5] has been proposed to link tasks to data: it consists in describing with textual annotations the operations that tasks perform on data stored in a database. Moreover, data are modeled with a UML class diagram.

An artifact-centric notation like GSM (Guard Stage Milestone) [6] stresses artifacts, which integrate business entities and life cycles. A life cycle is a process that defines the evolution of a business entity through stages: the processing of entities is done through tasks and transitions between stages through rules. The approach is flexible but lacks an overview of the dataflow.

PHILharmonicFlows [7] presents a notation based on micro-processes and macro-processes. The former represent the life cycles of business entities, the latter are used to orchestrate the evolution of life cycles. However, the dataflow is not represented.

The involvement of participants in a process is emphasized by the CMMN (Case Management Model and Notation) standard [8], which enables case workers to decide the order of execution of tasks and to assign tasks to each other. Data is not represented and neither is the dataflow.

In a comparison of notations based on their data modeling capabilities, the authors conclude that most practitioners consider data-centric notations more complicated than activity-centric ones and therefore data-centric approaches need further research [9].

A previous version of B2BPN has been illustrated [10] with reference to Cyber-Physical Systems (CPSs) [11]. The main purpose of CPSs is the integration of devices and software applications. That paper focused on the management of a manufacturing cell consisting of various machine tools, a warehouse, and a number of AGVs (Automated Guided Vehicles). The main aim was to handle the interactions between the cell control system and the devices; this perspective is different from a B2B one, although some contact points can be found, as will be discussed in the conclusion.

3. The feature of B2BPN

The notation is illustrated in this section with the help of a B2B process, the details of which are kept to a minimum.

The OrderHandling process acts in a distributor company that interacts with clients and suppliers. The interactions are based on the protocols that follow.

The distributor receives purchase orders from clients: the orders can be accepted or rejected. In the first case, the distributor informs the client when the order has been fulfilled. The distributor sends purchase orders to suppliers and is informed when the orders have been fulfilled.

An order has a unique identifier and contains n lines each of which indicates a type of product. Product types are known to all stakeholders, and have unique identifiers. Such data are called global.

The OrderHandling process deals with various types of products without stock. It receives purchase orders from clients and sends purchase orders to suppliers. The types of client orders and supplier orders are COrder and SOrder, respectively. The process does not generate a supplier order for each client order but can aggregate different lines of client orders into the same supplier order. When a supplier has fulfilled an order, this does not mean that the process is able to fulfill the order of a given client: to do so it must wait until all the supplier orders that contain the lines of the client order have been fulfilled.

In addition to the entities that a process manages, there are other entities that are not generated nor modified by the process. Such entities are called contextual. Such a distinction is in agreement with the Domain Driven Approach [12].

Client orders, supplier orders and lines are entities managed by process OrderHandling. On the other hand, product types, clients, suppliers and relationships between suppliers and product types are contextual elements. The roles of participants in the process (such as the Account Manager role) are contextual elements as well.

The behavior of the process is as follows.

A client order (COrder) can be accepted or rejected by the account manager associated with the client. He or she then chooses between two tasks, accepts or rejects. Both tasks cause an interaction with the client.

The output of the acceptance task is made up of the order lines, which are handled by the account manager as follows. He or she can aggregate lines into a new supplier order (which takes the pending state), can aggregate lines into a pending order, or can close a pending order, which then takes the closed state. A supplier order is composed of lines like a client order.

A closed order is sent to the supplier company, which then notifies the fulfillment of the order.

The output of this notification does not concern the order but the associated lines, which will assume the served state. When the served lines contain all the lines associated with a given client order, that order becomes served and the process informs the client that the order has been fulfilled.

The four types of models included in B2BPN are illustrated in the following subsections with reference to the above-mentioned B2B process.

3.1. Architectural model

The architectural model shows the reference company and its partners, along with the intended collaborations. The model for the B2B example is shown in Fig.1. The reference company is underlined. The collaborations are represented by ovals and their names consist of the abbreviation of the companies involved followed by the abbreviation indicating the purpose of the collaboration; PO means purchase order. The notation takes advantage of the SoaML specification [13].

The arrow leading into a collaboration icon shows the initiator of the collaboration; for example, a CD-PO collaboration is initiated by a client.

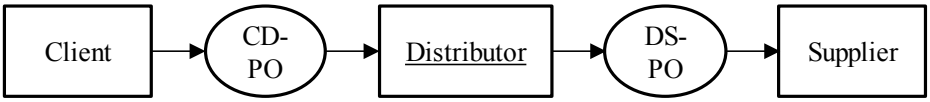

Fig.1. Architectural model of the B2B example.

3.2. Collaboration models

Collaboration models take their cue from UML interaction models [14] and extend them by adding information models related to the information transported by the interactions. In Fig.2, the above-mentioned collaborations are shown.

The interactions are represented with oriented arcs from the sender to the receiver. An interaction has the name of the type of data transmitted, possibly followed by the state of the data. If the state is missing, it is a new data. The message payload appears next to the interaction.

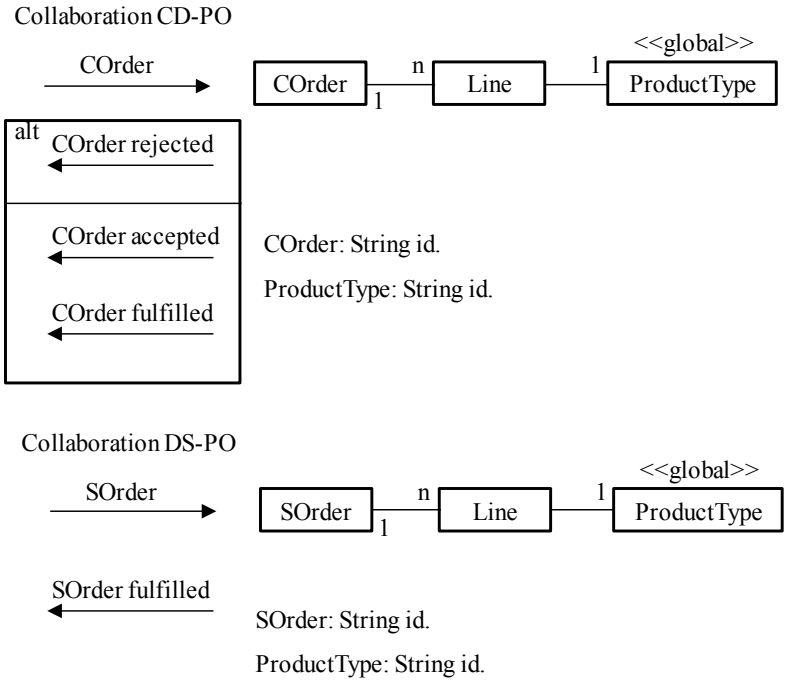

Fig.2. Collaboration models of the B2B example.

The first interaction of CD-PO brings a new client order: the payload shows that the client order is linked to n lines each of which points to a product type, which is a global information as indicated by the global stereotype. The attributes are shown separately.

A collaboration model is a sequence of interactions or blocks containing interactions and/or other blocks. Blocks can contain optional, repetitive or alternative subsequences: the corresponding indicators are opt, loop and alt. The CD-PO collaboration presents a block of alternative subsequences.

3.3. Process Information model

The information model of the OrderHandling process is shown in Fig.3. It is a UML class diagram [14].

The ProductType, Client, Supplier and AccountMgr classes are contextual while the others are managed by the process. The relationships between contextual classes, such as Client – AccountMgr and ProductType – Supplier, are also contextual.

The relationship arrows indicate mandatory relationships: if a class, say, A has a mandatory relationship with a class, say, B, the link between A and B has an arrow pointing to B.

For example, the generation of a COrder implies that the newly generated order is connected to n new lines and each of them is connected to a product type.

For simplicity, the attributes of classes are omitted in Fig.3.

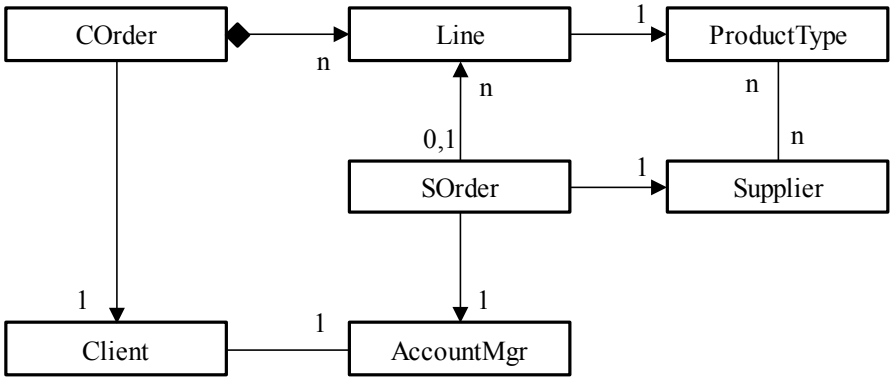

Fig.3. Information model of process OrderHandling.

3.4. B2B Process model

The model of the OrderHandling process is shown in Fig.4. The main elements are the tasks and the dataflow.

In the following description, the name of an entity type with the lowercase initial, e.g., cOrder, is used to indicate an entity of that type.

Tasks are divided into interaction tasks, (internal) tasks, and choices; they are represented by rectangles.

The notation is meant to indicate the effects of tasks in a declarative way by means of post-conditions; the constraints on the choice of the input entities are given by pre-conditions. For lack of space, these conditions are not shown formally but are mentioned in the task descriptions.

The interaction tasks process the corresponding interactions. They are placed in rectangles with rounded corners that represent collaborations: the name of the collaboration appears at the top of the rectangle. An input interaction task has one output, and an output interaction task has one input.

An internal task can be executed by a process participant (human task) or automatically by the process; in the latter case its name is preceded by “a:”. A human task is associated with a role and therefore the performer is a person who plays that role. Often, the performer is associated with the input entities and then he or she is predetermined; if this

association is missing, any person performing the required role can execute the task. For simplicity, the details of the determination of performers is omitted.

The choices are groupings of tasks that have the inputs in common: the choice is of the appropriate task to handle the inputs. A choice has a name and may be automatic.

The dataflow shows the types and states of the entities circulating in the process. It consists of direct links between tasks or indirect links based on places (represented by circles). Places are used to represent confluences (two or more inputs) and/or ramifications (two or more outputs), or when the entities are subject to choices.

As shown in Fig.4, the dataflow consists of three types of flows: those containing client orders (COrder initial, COrder accepted, COrder rejected, COrder fulfilled), those containing supplier orders (SOrder pending, SOrder closed) and those containg lines (Line pending, Line served). The flows are determined by the tasks.

Tasks have the semantics of Petri net transitions: they take entities from all inputs and add entities to all outputs. A task that has two or more inputs and one output is a join task, and a task having two or more outputs and one input is a fork task. A task is a join/fork task if it has two or more inputs and two or more outputs.

Stateless input interaction tasks (such as COrder) enter new entities into the dataflow. The COrder task generates a new cOrder based on the interaction payload and enters it into the dataflow with the initial state; the new entity is subject to the handleCOrder choice.

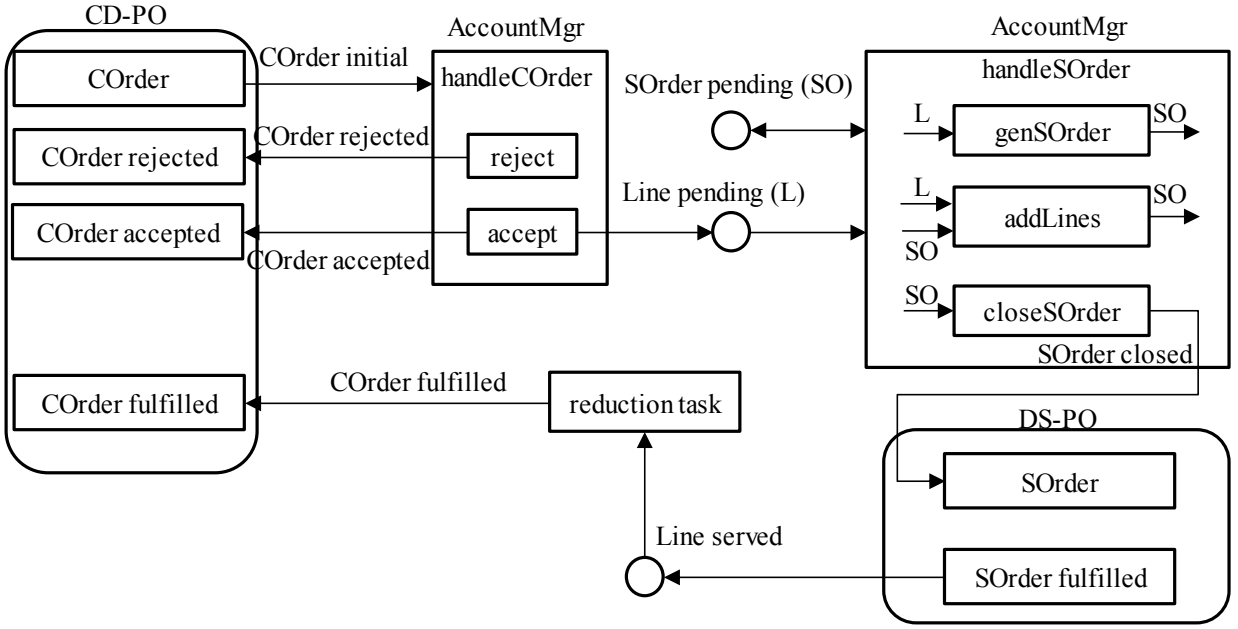

Fig.4. The model of process OrderHandling.

The reject task changes the state of the input entity from initial to rejected. The accept task is a fork one: it changes the state of the input entity from initial to accepted and emits the lines associated with the input entity thus carrying out a mapping from COrder to Line.

The handleSOrder choice contains three tasks. With the genSOrder task, the performer generates a new sOrder and chooses lines to aggregate into it; the result is a new sOrder in the pending state. Lines are taken from place Line pending: the short name of the place, i.e. L, is shown between parentheses. Short names are used to indicate the input and output places of the tasks included in the choice. The output of task genSOrder is put into place SOrder pending, whose short name is SO. With the addLines task, the performer chooses a pending order and some lines to aggregate

into it; the pending order is put back into place SOrder pending and the lines are removed from place Line pending. With the closeSOrder task, the performer chooses a pending task and changes its state from pending to closed.

The addLines task is a join one: the performer carries out a match between n pending lines and a pending sOrder: this choice is subject to the condition that the lines point to the product types handled by the supplier to which the order is addressed.

The process continues with the SOrder interaction task, which sends the order to the supplier. The supplier then responds with the interaction SOrder fulfilled and the output dataflow of the corresponding interaction task enters the lines associated with the order into place Line served, thus carrying out a mapping from SOrder to Line.

When place Line served contains all the lines related to the same cOrder, such lines must be removed from the place and the cOrder must get the state fulfilled. These operations are made by an automatic reduction task, which has only one input and only one output: the types of the input and output are different but there is a relationship many to one from the input type and the output one. When the input place contains all the entities linked to an entity whose type is the output type of the reduction task, the reduction takes place. The cOrder is put into the fulfilled state and the process passes it to the output interaction task COrder fulfilled.

4. Conclusion

This paper has presented a notation to model business processes in the context of B2B systems. In addition to process models, the notation (B2BPN) includes three other types of models, i.e. the architectural model, the collaboration models and the information model of the process.

The major features of the process model are as follows. The dataflow and the tasks are the main elements, the process is singleton (there is only one instance) and this gives great flexibility to compose and decompose data flows by means of join, fork, join/fork, and reduction tasks. In addition, the notation supports choices of tasks and of input entities. The choice of a task takes place when there is a group of tasks having inputs in common: the choice may be automatic or may be the outcome of the decision of the participant in charge of the choice. The choice of the input entities takes place with join tasks (which have two or more inputs) when a match of the entities taken from the inputs is required.

As to the future work, the main aim is to apply the B2B perspective to Cyber-Physical Systems (CPSs) by leveraging the notion of digital twins [15]. The advent of IoT (Internet of Things) [16] has underlined the need to integrate devices and software applications, but this is not a straightforward task as they are heterogeneous.

Digital twins can play an important role if they can incorporate processes able to interact with sensors and actuators of devices as well as with software applications.

Acknowledgements

The author thanks the reviewers for their helpful suggestions.

References

[1] BPMN, Business Process Model and Notation. Retrieved April 16, 2020, from https://www.omg.org/spec/BPMN/.

[2] Hohpe, G., Woolf, B. (2004) “Enterprise Integration Patterns”. Addison-Wesley.

[3] Meyer, A., Pufahl, L., Fahland D., Weske M. (2013) “Modeling and enacting complex data dependencies in business processes”. In: 11th International Conference on Business Process Management, LNCS, vol. 8094, pp. 171–186, Springer, Heidelberg.

[4] Sanz, J.L.C. (2011) “Entity-centric operations modeling for business process management - A multidisciplinary review of the state-of-the-art”. In: 6th IEEE International Symposium on Service Oriented System Engineering, pp. 152–163,.

[5] Combi, C., Oliboni, B., Weske, M., Zerbato, F. (2018) “Conceptual modeling of inter-dependencies between processes and data”. In: 33rd ACM Symposium on Applied Computing, pp. 110–119.

[6] Hull, R. et al. (2010) “Introducing the Guard-Stage-Milestone approach for specifying business entity lifecycles”. In: 7th International Workshop on Web Services and Formal Methods, LNCS, vol. 6551, pp. 1–24. Springer, Heidelberg.

[7] Künzle, V., Reichert, M. (2011) “PHILharmonicFlows: towards a framework for object-aware process management”. Journal of Software Maintenance and Evolution: Research and Practice, 23(4), 205–244.
[8] CMMN, Case Management Model and Notation. Retrieved April 16, 2020, from https://www.omg.org/spec/CMMN/1.1/.
[9] Steinau, S., Marrella, A., Andrews, K., Leotta F., Mecella, M., Reichert, M. (2019) “DALEC: a framework for the systematic evaluation of data-centric approaches to process management software”. Software & Systems Modeling, 18, 2679–2716.
[10] Bruno, G. (2019) “A modeling approach for Cyber-Physical Systems based on collaborative processes”. IFAC PapersOnLine 52-13, 2764–2769.
[11] Lee, E.A. (2008) “Cyber physical systems: design challenges”. In: 11th IEEE Symposium on Object Oriented Real-Time Distributed Computing, pp. 363–369.
[12] Vernon, V. (2013) “Implementing Domain-Driven Design”. Addison-Wesley.
[13] SoaML, Service oriented architecture Modeling Language. Retrieved April 16, 2020, from https://www.omg.org/spec/SoaML/About-SoaML/.
[14] UML, Unified Modeling Language. Retrieved April 16, 2020, from https://www.omg.org/spec/UML/About-UML/.
[15] Rasheed, A., San, O., Kvamsdal, T. (2020) “Digital twin: values, challenges and enablers from a modeling perspective”. IEEE Access 8, 21980–22012.
[16] Atzori, L., Iera, A., Morabito, G. (2010) “The Internet of things: a survey”. Computer Networks 54, 2787–2805.