Data flow and human tasks in business process models

Available online at www.sciencedirect.com

ScienceDirect

Procedia Computer Science 64 (2015) 379 – 386

Procedia Computer Science

Conference on ENTERprise Information Systems / International Conference on Project MANagement / Conference on Health and Social Care Information Systems and Technologies, CENTERIS / ProjMAN / HCist 2015 October 7-9, 2015

Data flow and human tasks in business process models

Giorgio Bruno*

Politecnico di Torino, Torino, Italy
giorgio.bruno.@polito.it

Abstract

In contrast with the traditional view that represents business processes as flow charts of tasks, the artifact-centric one stresses the importance of the data flow, as the main responsible for the activation of the tasks. This viewpoint leads to reconsider the interactions between the process and its tasks as well as the execution mode of the tasks. The greatest benefits concern human tasks; they should no longer be considered only as services implemented by people but they may enable their performers to make choices. Two kinds of human choices are considered in this paper: the choice of the inputs to be acted on, and the choice of the course of action to be taken. The execution mode of human tasks is also examined and three categories are illustrated: performer-driven tasks, process-driven tasks and macro tasks. These categories come with a number of patterns, which are exemplified in this paper.

Keywords: business processes; artifacts; human tasks; data flow

1. Introduction

The limitations of the traditional view that represents business processes as flow charts of tasks have been overcome owing to the introduction of the artifact-centric approach, which shifts the focus to the business entities involved in the processes. Since this perspective has its roots in past research on entity-based dynamic modeling[15], it is particularly significant in the domain of information systems, whose purpose is to enable human participants and/or machines to perform work using information[1].

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

doi:10.1016/j.procs.2015.08.502

While the traditional view emphasizes the control flow, the artifact-centric one stresses the importance of the data flow, as the main responsible for the activation of the tasks. The data flow shows the types and the states of the input and the output entities of the tasks, and therefore it combines the life cycles of the entities involved in the process.

The emphasis given to the data flow leads to reconsider the interactions between the process and its tasks as well as the execution mode of the tasks. The greatest benefits concern human tasks. In fact, in the traditional view, the participants in the process are considered as mere resources needed to carry out tasks that are not automatable. Such tasks are activated through a request-response protocol: performers receive inputs and return results. If human choices are needed, they are not clearly represented in the process model. In the WS-HumanTask[20] standard, human tasks are considered as services implemented by people; this interpretation allows for a simple integration of persons in service-oriented applications, but it does not provide high-level constructs for human choices.

In this paper, two kinds of human choices are considered: the choice of the inputs to be acted on, and the choice of the course of action to be taken. An example of the first category is the choice of a number of requisition orders coming from customers so as to produce a procurement order directed to a supplier. The implied human activity is not meant to reply to each requisition order with a procurement order; on the contrary, it must be fed with the incoming requisition orders and must be able to reply with procurement orders asynchronously.

An example of the second category is the decision that an account manager may take about an incoming request for quote; they may: reject the request, provide the quote if they have the resources required, or forward the request to a partner. Three courses of action are then possible and the decision is up to the account manager.

The organization of this paper is as follows. Section 2 is about the related work; section 3 gives an overview of the notation used in this paper and introduces three categories of human tasks which will be thoroughly examined in sections 4 to 6. Then, section 7 presents the conclusion.

2. Related work

The research reported in this paper addresses knowledge-intensive business processes[6] and is grounded on recent work on the artifact-centric perspective[8] and on flexible processes[16]. Artifacts denote concrete and self-defining chunks of information used to run a business[2,5,13]. The notion of artifact type includes both informational aspects and dynamic ones; the latter are defined by a life cycle, which shows how the actual entities evolve over time through states and transitions.

The artifact-orientation is put into practice in two major ways; life cycles may be defined separately or may be combined in one model. In the first case, the life cycles interact in a number of ways: events and rules in the Guard-Stage-Milestone approach[9], macro processes in PHILharmonicFlows[10], hierarchical relationships in COREPRO[12], and messages in Proclets[18].

In the second case, the combination of the life cycles gives rise to a data flow which shows the types and the states of the input and output entities of the tasks. This approach may also be applied to well-known activity-centric notations such as BPMN[3] and UML activity diagrams[17]. The drawback is the coexistence of the control flow and the data flow: this makes it difficult to handle a number of situations, such as the selection of homogeneous entities to be processed in batches and the many-to-many mapping between entities of different types, such as requisition orders and procurement orders[11]. The difficulty has been put in relation[14] to the weakness of the notion of process instance, which is avoided in the notation used in this paper.

On the contrary, the notation adopted in this paper makes the activation of the tasks depend only on the data flow and this allows for a better representation of human choices.

Human choices have been addressed by research on flexible processes based on declarative approaches[7]. For example, in Declare[19] any task of an ad-hoc sub-process may be performed as long as the mandatory constraints are not violated. Such sub-processes bear some similarities with the macro tasks presented in this paper; however, since they do not explicitly represent the data flow, the choice of the tasks cannot be put in relation with the input data.

The interactions between processes and human tasks may be analyzed from several points of view. In the WS-HumanTask[20] standard, they are considered as services implemented by people; this view allows for a simple integration of persons in service-oriented applications, but it does not provide high-level constructs for human choices, such as the ones illustrated in this paper.

3. Overview

The notation for business processes proposed in this paper is a network of tasks interconnected by the data flow[4]. Tasks represent units of work that act on the input entities so as to provide the output ones. The precedence between tasks is determined by the data flow, since the input entities of a task are the output ones of other tasks. The notion of process instance is not needed in that all the information is contained in the entities forming the data flow: process variables, which are the main reason for the introduction of process instances, are not required.

An example of process is given in Fig.1. The process represents a fragment of a conference management system, which may be described as follows. The chairman of a forthcoming conference enters a conference entity containing the information about the conference, e.g. the time limit for submitting papers (Date d1). Then, authors are entitled to submit papers before d1; in addition, they can update and also withdraw the papers submitted. After d1, the chairman assigns the papers to the referees.

All the tasks in Fig.1 are human tasks. A human task pertains to a given role: only the members of the role are allowed to perform instances of the task. The role name, such as Chairman, appears near the task symbol.

A task instance may be anticipated by the process or may be activated by the performer when they want to. In the first case, the task instance appears in the work list of the intended performer. In the second case, it is the performer who determines the instance; as a consequence, the task is activated from a menu.

The two cases will be referred to as process-driven or performer-driven, respectively. Such qualifiers may also be applied to tasks; hence, a performer-driven task is a task whose instances are performer-driven.

The icon for a performer-driven task is a rectangle with rounded corners; the icon for a process-driven task is an ordinary rectangle. Tasks acting on the same entities are grouped in macro tasks, such as handlePaper. Such tasks indicate alternative courses of action and the choice of the task is up to the performer of the macro task. The input link of task updatePaper is an undirected arc because this task may be performed several times; the undirected arc means that the input entity remains in the input flow. On the contrary, task withdrawPaper removes the paper from the input flow.

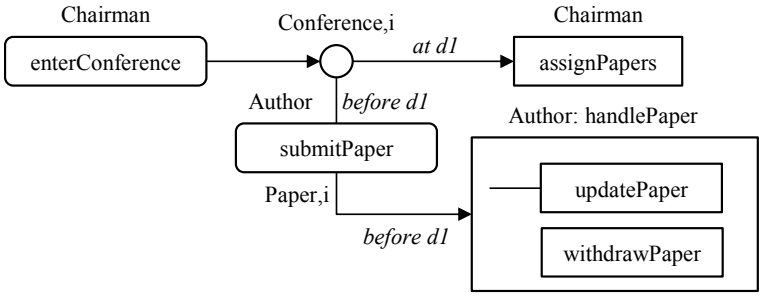


Fig.1. Example of business process

The data flow is made up of the input and output links of tasks; each link has a label showing the type and the state of the entities flowing through it. Junctions and branches are represented with circles.

Performer-driven tasks, process-driven tasks and macro tasks are illustrated in the next sections.

4. Performer-driven tasks

The tasks included in this category enable persons to participate in processes; their performers become known to the processes and will be involved in subsequent process-driven tasks. Some examples are shown in Fig.2. There are three major patterns: generative tasks, associative tasks and acquisitive ones.

A generative task starts a path in the process being considered: it has no input and is meant to generate a new entity, which will enter the initial state of its life cycle. Task enterConference may be performed by any chairman; the result is a new Conference entity in the initial state (named "i").

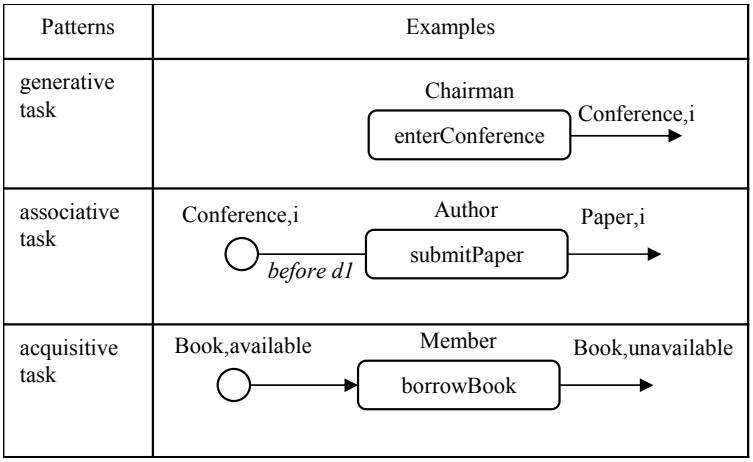


Fig.2. Patterns of performer-driven tasks

Task submitPaper enables authors to submit papers to conferences. The execution of the task implies the selection of a conference in the initial state and it results in the generation of a Paper entity related to the conference and to the author. It is an associative task in that it does not acquire the conference entity; it simply links a new paper entity to the conference entity. The associative feature is shown by the undirected arc between the conference state and the task. A number of authors may submit papers to the same conference at the same time. Usually such tasks have time limits; in the example, the time limit is contained in attribute d1 (a date) of the conference entity.

An acquisitive task, such as borrowBook, enables the performer to select an entity from among those available and to take it out of the input flow.

5. Process-driven tasks

The instances of process-driven tasks appear in the work list of the intended performers. Clicking on a work list item implies performing the corresponding activity: the input entities are predetermined by the process. However, there are cases in which the performer may select the entities to be acted on from among those predetermined by the process.

Four patterns for process-driven task instances are considered in this paper; they are presented in Fig.3.

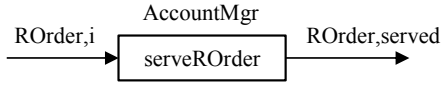
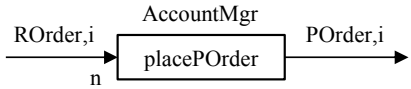
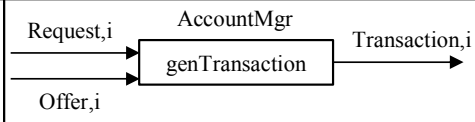
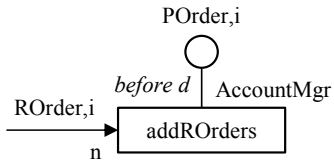
Patterns	Work list items	Examples
modificative task	serveROrder, ROrder	
aggregative task	placePOrder, ROrders	
matching task	genTransaction, Requests, Offers	
mixed matching task	addROrder, ROrders, POrders	

Fig.3. Patterns of process-driven tasks

The "Work list items" column shows the content of the corresponding work list item in an abstract form consisting of the task name and some information about the entities to be acted on. A singular name, such as ROrder, denotes the information about a specific entity. A plural name, such as ROrders, stands for a collection of entities: what is shown in the actual work list item is their number.

The first pattern (modificative task) addresses tasks having one input flow and operating on one input entity at a time; for each input entity, a task instance appears in the work list of the performer. In the example, an account manager serves the incoming requisition orders (whose type is ROrder) one at a time.

In general, an input entity is assigned to a specific performer (i.e., a specific participant playing the role required by the task) on the basis of a rule, called performer rule. For simplicity, performer rules are not included in the examples; for task serveROrder, the performer rule assigns the requisition order to the account manager taking care of the relations with the customer who placed the requisition order.

An aggregative task has one input flow and acts on a number of input entities: the choice of the input entities from among those available is up to the performer. Only one instance of the task appears in the work list; in fact, when the performer clicks on the work list item, the activity associated with the task enables them to select the input entities from among those predetermined by the process on the basis of the corresponding performer rule. In the example, task placePOrder enables an account manager to combine several requisition orders into one procurement order (whose type is POrder). The aggregation is indicated by weight "n" on the input flow.

A matching task has two or more input flows and the performer is in charge of selecting the entities to be taken from the flows. The number of entities to be taken from each flow may be specified by putting weights on the corresponding arcs: the default number is one. In the example, task genTransaction produces a commercial transaction out of one request and one offer. A pre-condition may be added to the task so as to set a constraint to the matching.

In the mixed matching pattern, not all the inputs are predetermined by the process: there is at least one associative flow and the other flows are modificative or aggregative. The associative flows (represented by undirected arcs) indicate that the choice of the corresponding inputs is made by the performer during the execution of the task. In the example, task addROrders adds a number of requisition orders to the procurement order selected by the performer.

Several performers may add requisition orders to the same procurement order, in that it is not acquired by the task instances. The POrder input flow is an associative flow, and as such it has a time limit; in the example, the time limit is contained in attribute d (a date) of the POrder entities.

6. Choices

It often happens that the same input entity may be handled in different ways each resulting in a different course of action, and the choice of the option is a human decision. For example, an account manager may reject incoming requisition orders or may combine a number of them in one procurement order. If two task instances were put in the work list, they should be mutually exclusive. To avoid this, the notion of macro task is introduced. A macro task enables the performer to select the task with which to handle the input entities, if two or more options are feasible. The graphical representation is a box including the options; the name of the macro task is shown after the role name.

Four patterns are presented in Fig.4: the first two patterns refer to simple choices and the others to complex ones.

Usually a simple choice is a macro task with one input flow. In the example of the first pattern, macro task decideAboutROrders enables an account manager to reject requisition orders or to combine them in procurement orders.

A special case of simple choice is the optional choice. In the example, macro task decideAboutPaper allows an author to withdraw a paper submitted to a conference, provided that the task is carried out before the time limit specified in attribute d1 of the conference entity the paper is associated with; otherwise, the paper is automatically moved to state “submitted”.

A simple choice may have more than one input flow: in that case, each option concerns all the input flows. On the contrary, a complex choice has two or more input flows and the options may relate to subsets of the input flows. What is important is that each input flow is handled by at least one option and that the options cannot be divided into two partitions having no common input flows.

Complex choices come in two major flavors, depending on whether the input entities are interrelated or not.

Macro task handleR&O enables an account manager to generate a transaction out of a request and an offer, but, in addition, requests and offers may be rejected independently. The inputs are given aliases, such as “r” and “o”, to be used to indicate which inputs each task is meant to handle. The inputs are unrelated in the sense that requests and offers are generated independently from each other.

On the contrary, macro task handleCRfQ deals with inputs that are interrelated. Its purpose is to enable account managers to provide quotes in reply to requests for quote coming from customers; to do so they first have to get quotes from suppliers and then they can produce a customer quote on the basis of the supplier quote selected from among those received. CRfQ and SRfQ are the types of the requests for quote coming from customers and of those directed to suppliers, respectively; CQuote and SQuote are the types of the quotes directed to customers and of those received from suppliers, respectively. The types of the inputs of the macro task are CRfQ and SQuote; their aliases are “r” and “sq”, respectively.

Task enterSRfQ inserts a supplier rfq (request for quote), which is linked to the customer rfq that caused its generation. Suppliers reply to requests for quote with quotes; a supplier quote is related to the supplier rfq it is a reply to, and then it is indirectly linked to the customer rfq that caused the supplier rfq to be issued. The input of task enterSRfQ is shown with an undirected arc in that the task does not consume the input rfq. The reason is that it may be repeated several times: in fact, an account manager may issue several requests over time, in case the quotes received are not satisfactory.

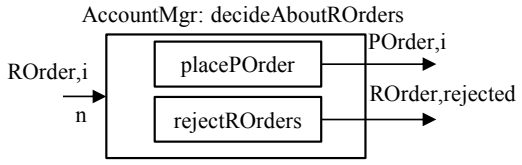
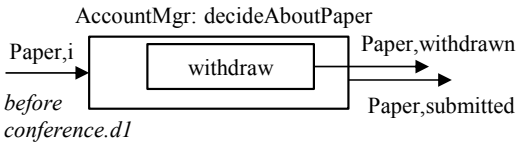
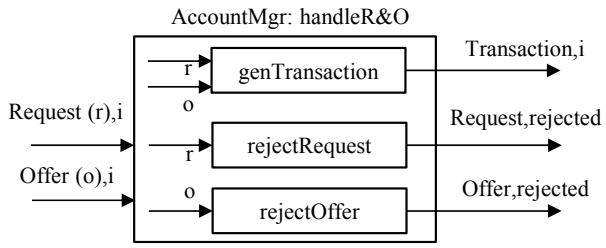
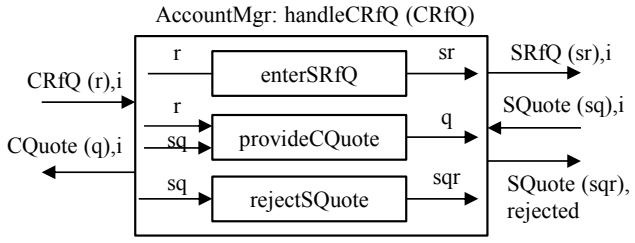
Patterns	Work list items	Examples
simple choice	decideAbout ROrder, ROrders	
optional choice	decideAbout Paper, Paper	
complex choice: unrelated inputs	handleR&O, Requests, Offers	
complex choice: related inputs	handleCRfQ, CRfQ, SQuotes	

Fig.4. Examples of choices

Task provideCQuote enables the account manager to select one of the supplier quotes associated with the customer rfq so as to produce a quote for the customer. The inputs of the macro task are interrelated because a supplier quote is necessarily associated (through a supplier rfq) with a customer rfq; a customer rfq cannot be matched with an arbitrary supplier quote but only with one of those associated with it. The customer rfq is the dominant entity in the matching and then type CRfQ dominates type SQuote. The dominant types are shown between parentheses after the name of the macro task: this is a syntactical difference which allows distinguishing complex choices with related inputs from those with unrelated inputs.

The supplier quotes to be rejected are handled by task rejectSQuote. The macro task is completed when a customer quote has been produced and all the supplier quotes have been handled.

7. Conclusion

This paper considers a business process model as a network of tasks interconnected by a data flow. Tasks represent units of work that receive input entities from the data flow and return output entities to it. This viewpoint

gives more freedom to the persons participating in the process; they are no longer considered as mere resources needed to carry out tasks that are not automatable, but they are actively involved in the choice of the inputs to be acted on, and on the choice of the course of action to be taken. These choices are carried out through three categories of tasks, which are referred to as performer-driven tasks, process-driven tasks and macro tasks. Performer-driven tasks are subject to the will of their performers and hence they are activated from a menu. On the contrary, the other kinds of tasks are anticipated by the process and the participants may find instances of these tasks in their work lists.

Current work is being devoted to the definition and implementation of work lists able to cope with the task patterns illustrated in this paper.

Acknowledgements

The author wishes to thank the anonymous reviewers for their helpful comments.

References

1. Alter, S.: Defining information systems as work systems: implications for the IS field. European Journal of Information Systems, 17, 448–469 (2008)
2. Bhattacharya, K., et al.: A model-driven approach to industrializing discovery processes in pharmaceutical research. IBM Systems Journal, 44 (1), 145–162 (2005)
3. BPMN, Business Process Model and Notation, V.2.0.2. Retrieved March 30, 2015, from http://www.omg.org/spec/BPMN/2.0.2/
4. Bruno, G.: A data-flow language for business process models. Procedia Technology 16, 128–137 (2014)
5. Chao, T., et al.: Artifact-based transformation of IBM Global Financing. LNCS, vol. 5701, pp. 261–277. Springer, Heidelberg (2009)
6. Di Ciccio, C., Marrella, A., Russo, A.: Knowledge-intensive processes: characteristics, requirements and analysis of contemporary approaches. Journal on Data Semantics, 4, 29–57 (2015)
7. Goedertier, S., Vanthienen, J., Caron, F.: Declarative business process modelling: principles and modelling languages. Enterprise Information Systems, 2, 161–185 (2015)
8. Hull, R.: Artifact-centric business process models: brief survey of research results and challenges. LNCS, vol. 5332, pp 1152–1163. Springer, Heidelberg (2008)
9. Hull, R., et al.: Introducing the Guard-Stage-Milestone approach for specifying business entity lifecycles. LNCS, vol. 6551, pp. 1–24. Springer, Heidelberg (2011)
10. Künzle, V., Reichert, M.: PHILharmonicFlows: towards a framework for object-aware process management. Journal of Software Maintenance and Evolution: Research and Practice, 23 (4), 205–244 (2011)
11. Meyer, A., Pufahl, L., Fahland, D., Weske, M.: Modeling and enacting complex data dependencies in business processes. LNCS, vol. 8094, pp. 171–186. Springer, Heidelberg (2013)
12. Müller, D., Reichert, M., Herbst, J: Data-driven modeling and coordination of large process structures. LNCS, vol. 4803, pp. 131–149. Springer, Heidelberg (2007)
13. Nigam, A., Caswell, N.S.: Business artifacts: an approach to operational specification. IBM Systems Journal, 42(3), 428–445 (2003)
14. Sadiq, S., Orlowska, M., Sadiq, W., Schulz, K.: When workflows will not deliver: the case of contradicting work practice. In 8th Int. Conference on Business Information Systems (2005)
15. Sanz, J.L.C.: Entity-centric operations modeling for business process management - A multidisciplinary review of the state-of-the-art. In: 6th IEEE Int. Symposium on Service Oriented System Engineering, pp. 152–163. IEEE Press, New York (2011)
16. Schonenberg, H., Mans, R., Russell, N., Mulyar, N., van der Aalst, W.: Process flexibility: A survey of contemporary approaches. LNBIP, vol. 10, pp. 16–30. Springer, Heidelberg (2008)
17. UML. Unified Modeling Language, V.2.4.1. Retrieved March 30, 2015, from http://www.omg.org/spec/UML/2.4.1/
18. van der Aalst, W.M.P., Barthelmess, P., Ellis, C.A., Wainer, J.: Workflow modeling using Proclets. LNCS, vol. 1901, pp. 198–209. Springer, Heidelberg (2000)
19. van Der Aalst, W. M., Pesic, M., Schonenberg, H.: Declarative workflows: balancing between flexibility and support. Computer Science-Research and Development, 23:2, 99-113 (2009)
20. Web Services Human Task (WS-HumanTask) V.1.1. Retrieved March 30, 2015, from http://docs.oasis-open.org/bpel4people/ws-humantask-1.1.html