A UML 2 Profile for Event Driven Process Chains [*]

Birgit Korherr and Beate List
Women's Postgraduate College for Internet Technologies
Institute of Software Technology and Interactive Systems
Vienna University of Technology
{korherr, list}@wit.tuwien.ac.at
http://wit.tuwien.ac.at

Abstract. The Event-driven Process Chain (EPCs) is a very well established business process modelling diagram. It can be used as the starting point for software development and therefore, software engineers have to read these diagrams but prefer a well-known notation. For that reason, we have developed a UML 2 profile for EPCs based on a 1 - 1 mapping with UML 2 Activity Diagrams. The profile is tested with an example business process.

1 Introduction

Event-driven Process Chains (EPCs) have become widely-used for business process modelling in organisations. EPCs are based on Petri nets and incorporate role concepts and data models like ER models or UML class diagrams. As EPCs and software developers share models, and because business processes represent business requirements in a formal notation, EPCs can be used as the starting point for software development. For example, an EPC can be utilised to elicit requirements for a new software system, or to check whether the functions of an existing software system match with the requirements of a new business process. But most software developers are not aware of EPCs or are not able to read the models, as different modelling languages are used in traditional software engineering. In order to overcome this gap, we have developed a UML 2 profile for EPCs, with the goal:

- To provide EPC models to software developers in a well-known notation.
- To present EPC models to software developers through UML tools.

UML profiles provide an extension mechanism for building UML models for particular domains. Surprisingly, none of the existing UML profiles for business process modelling [1], [3], [4], [6], [9] cover EPCs. Generally, well-established business process languages have never been considered for UML profiles before.

The UML 2 profile for EPCs is realised as a 1-1 mapping between EPCs and UML2 activity diagrams, as both diagram types provide similar concepts. The contribution of the UML 2 profile for EPCs is:

[*] This research has been funded by the Austrian Federal Ministry for Education, Science, and Culture, and the European Social Fund (ESF) under grant 31.963/46-VII/9/2002.

- EPCs are available in a lot of modelling tools. The profile facilitates the seamless integration of already available EPC process models into a UML tool, without any additional modelling effort. In contrast, business process modelling diagrams that have no sufficient tool support require modelling from scratch.
- It provides business process models to software developers in UML notation. As software systems support the business processes of an organisation, the profile represents business requirements to software developers in a formal and well-known modelling notation.
- The profile can support the elicitation of requirements from the business process models for the software systems to be developed. Deriving requirements from the business process models ensures a business-goal oriented software development.
- The profile can be integrated into the Computation Independent Model (CIM) of the Object Management Group's (OMG) Model Driven Architecture (MDA) approach. Because the CIM model is a business model capturing the requirements of the software systems and is traceable to code, the integration of the UML profile can improve the quality of the requirements and the design of the software. The profile is raising the level of abstraction at which software development starts.
- The UML 2 profile for EPCs can be easily extended and mapped to Business Process Execution Languages (BPEL). Mapping tools are able to take the business processes models developed in a UML tool and convert them to the correct BPEL, and vice versa. Thus, high productivity will be resulting, even if the underlying technology changes.
- The profile could abandon Business Process Modelling Tools, as almost all UML-tools support UML profiles.
- The profile integrates EPC models into the standard software development environment and can be seen as a further step towards bridging the gap between business process engineering and software engineering.

Based on the theoretical concepts of the EPC in Section 2, and its meta-model (Section 3), we have developed the UML 2 profile for EPCs described in Section 4. It is realised as a 1 - 1 mapping between EPCs and UML 2 activity diagrams. Each element of the EPC meta-model is described with stereotypes. For expressing constraints on the UML 2 profile for EPCs we use the Object Constraint Language. The UML 2 profile for EPCs is tested with an example business processes in Section 5. Related work is discussed in Section 6.

2 The Event-Driven Process Chain (EPC)

The EPC [8] has been developed within the framework of the Architecture of Integrated Information System (ARIS) and is used by many companies for modelling, analysing, and redesigning business processes. The EPC is based on the concepts of stochastic networks and Petri nets. EPCs are targeted to be easy understood and used by business people. The ARIS concept [8] divides complex processes models into separate views, in order to reduce the complexity. The views can be handled independently as well as related. There are three views focusing on functions, data, and the organisation (see Fig. 1), and an additional view focusing on their integration.

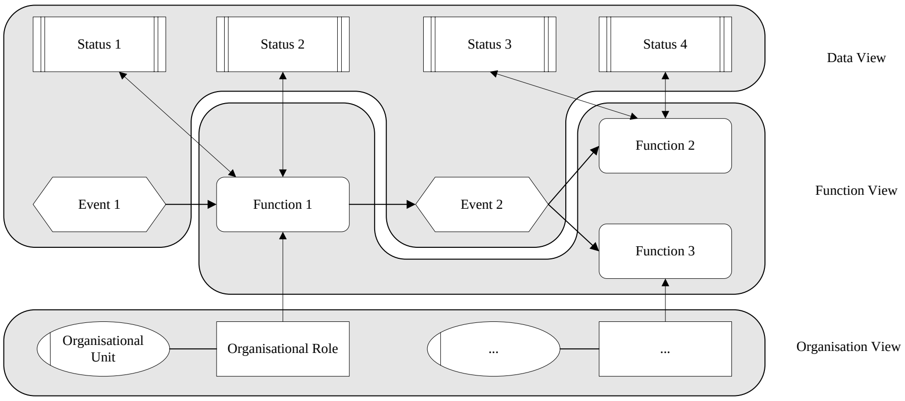


Fig. 1. Architecture of Integrated Information System (ARIS) Views

The *Data View* contains events and statuses. Events such as “customer order received” and statuses such as “article status” represent data. The *Function View* contains the description of the activities to be performed, the individual sub-functions, and relationships that exist between the functions. The *Organisation View* represents the organisational structure. This includes organisational units, employees and roles as well as their relationships. The *Control View* links functions, organisation and data. It integrates the design results, which were initially developed separately. The functions, events, information resources, and organisation units are connected by the control flow. The resulting model is the EPC. It consists of the following elements:

- *Functions* are active elements and model the activities within the company.
- *Events* are created by processing functions or by actors outside of the model. An event may act as a pre- or post-condition of a function.
- *Logical operators* connect functions and events. There are three types of logical operators: AND, XOR (exclusive or) and OR.
- The *Organisation Unit* or *Role* is responsible for performing a function.
- The *Information Objects* portray input data serving as the basis for a function, or output data produced by a function. They correspond to entities or attributes of Chen’s Entity-Relationship model or the UML class diagram.
- The *Deliverables* represent services or products functions produce or need.

3 Meta-Model of the Event-Driven Process Chain

As already mentioned, EPCs are widely-used for business process modelling in organisations, not only in the German speaking language area but meanwhile worldwide. The meta-model of the EPC is described in Fig. 2. Each EPC consists of one or more *Functions* and two or more *Events*, as an EPC starts and ends with an event and requires at least one function for describing a process. That means that a function has at least one successor and one predecessor. Both, functions and events can be (re)used in several EPCs. An event has 4 attributes, *start, intermediate, end* and *trigger*. Start, intermediate and end shows if the event is at the beginning, middle or end of a process. Trigger demonstrates if an event triggers a logical operator. A function can be either a *Complex Function* or an *Elementary Function* and may be connected with one or more *Additional Process Objects.* A complex function may be refined by one or more functions. A function is connected with two *Control Flow Connectors*. An event is connected with one or two control flow connectors, as events start the EPC. Control flows link events with functions, but also events or functions with *Logical Operators.* A logical operator can be either an *XOR, OR* or *AND*. It is connected at least with 3 control flows, one or more incoming as well as outgoing connectors.

An *Additional Process Object* may be assigned to one or more functions. A *Deliverable*, an *Information Object* and an *Organisational Structure* are called additional process objects. All three types of additional process objects may be assigned to one or more functions. The organisational structure can be an *Organisational Role* or an *Organisational Unit*, the latter is refined by one or more organisational roles. The organisational structure is connected with one or more *Organisational Flow Connectors*. The information object is connected with one or more *Data Flow Connectors* and the deliverable is connected with one or more *Input/Output Flow Connectors.*

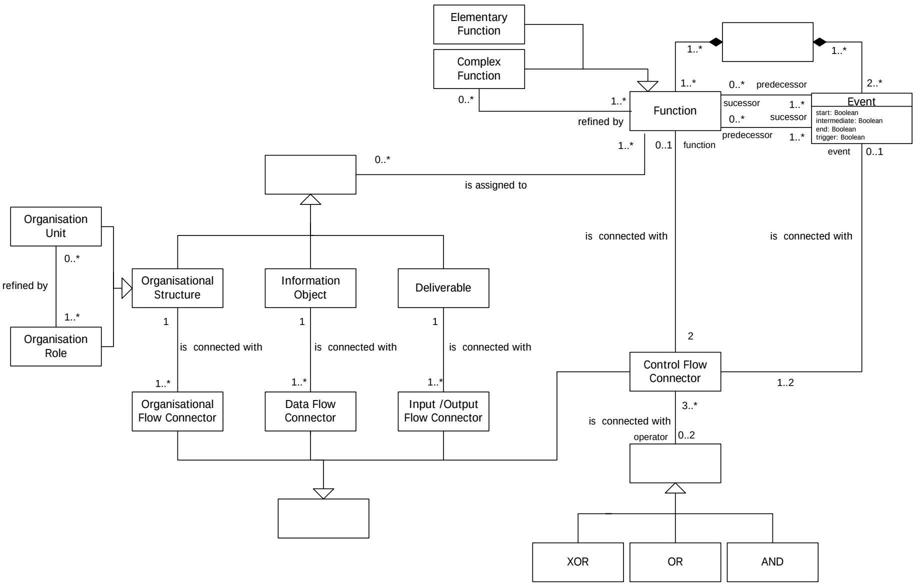


Fig. 2. Meta-Model of the Event-Driven Process Chain

4 The UML 2 Profile for EPCs

In this section we describe the UML 2 profile for EPCs. UML offers a possibility to extend and adapt its meta-model to a specific area of application through the creation of profiles. UML profiles are UML packages with the stereotype «profile». A profile can extend a meta-model or another profile [7] while preserving the syntax and semantic of existing UML elements. It adds elements which extend existing classes. UML profiles consist of *stereotypes*, *constraints* and *tagged values*.

The UML 2 profile for EPCs provides business process models in UML 2 notation. It can be used as the starting point for software development in order to achieve a better coordination between business processes and the supporting software systems. The profile is realised as a 1-1 mapping between EPCs and UML 2 activity diagrams, as both diagram types have similar concepts. In Table 1 to 4 every element of the EPC gets a proper element or base class of the UML 2 activity diagram assigned. For expressing constraints on the UML 2 profile for EPCs we use the Object Constraint Language (OCL) [5], described in Table 5.

Table 1. Mapping of EPC Functions and Events to UML 2 Elements

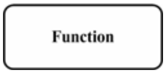
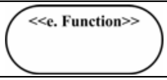
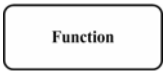
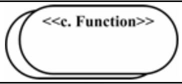
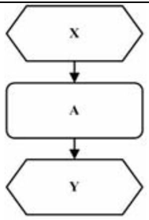
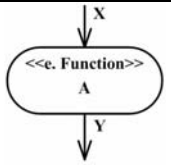
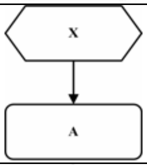
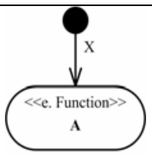
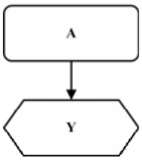
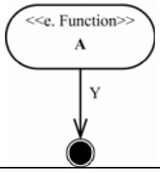
EPC Element	EPC Notation	UML 2 Base Class	UML Profile
Elementary Function		Action	
Complex Function		Action	
Event		Control Flow	
Start Event		Initial Node	
End Event		Final Node	

Table 2: Mapping of EPC Additional Process Objects to UML 2 Elements

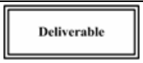
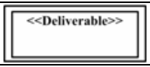
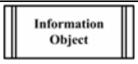
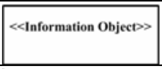
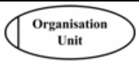
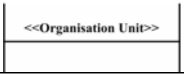
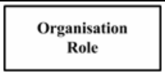
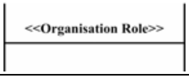
EPC Element	EPC Notation	UML 2 Base Class	UML Profile
Deliverable		Object	
Information Object		Data Store Node	
Organisation Unit		Activity Partition	
Organisation Role		Activity Partition	

Table 3. Mapping of EPC Flow Connectors to UML 2 Elements

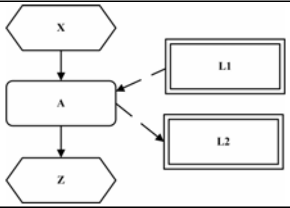
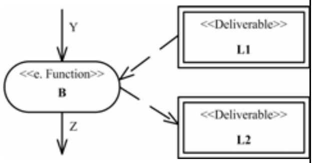
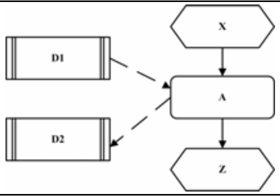
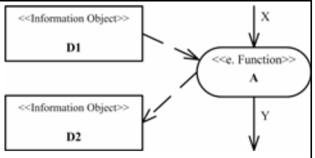
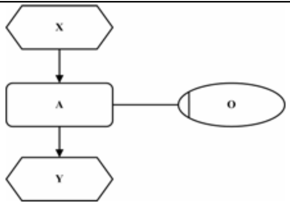
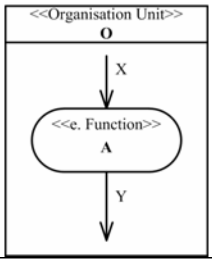
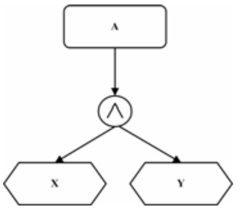
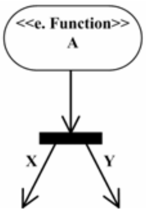
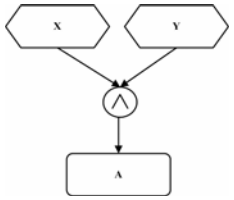
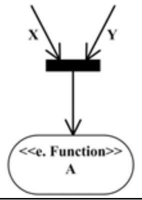
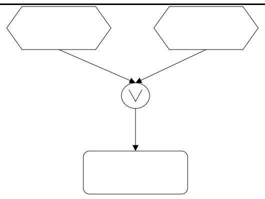
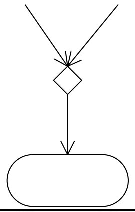
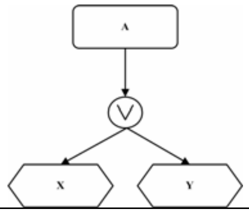
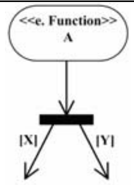
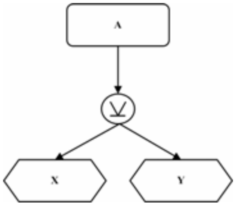
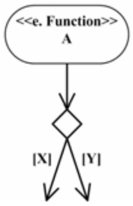
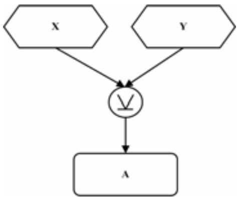
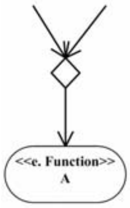
EPC Element	EPC Notation	UML 2 Base Class	UML profile
Input / Output Flow Connector		Object Flow	
Data Flow Connector		Object Flow	
Organisation Flow Connector		Activity Partition	

Table 4. Mapping of EPC Logical Operators to UML 2 Elements

EPC	EPC Notation	UML 2.0 Base Class	UML Profile
AND		Fork Node	
AND		Join Node	
OR		Merge Node	
OR		Fork Node with guards	
XOR		Decision Node (with Guards)	
XOR		Merge Node	

X Y

A

Table 5. Constraints of the UML 2 Profile for EPCs

UML 2 Base Class	EPC Element	Constraints

Action	Function	Before and after a function always happens an event.
		`context Function inv:`
		`self.Event[predecessor] and`
		`self.Event[successor] ->size() > 0`

Function	Complex Function	A complex function is refined by one ore more functions.
		`context ComplexFunction inv:`
		`self.Function->size() >= 1`

Control Flow	Event	An event is always the starting and the endpoint of the profile.
		context Event inv:
		`if Event.start = true and`
		`Event.intermediate = false and Event.end = false`
		`then function[predecessor] ->size() = 0`
		`endif`
		`if Event.intermediate = true and`
		`Event.start = false and Event.end = false`
		`then Function[predecessor] and`
		`Function[successor] size() > 0`
		`endif`
		`if Event.end = true and Event.start = false and`
		`Event.intermediate = false`
		`then Function[successor] ->size() = 0`
		`endif`
		An event is not allowed to trigger a xor-decision and or-fork.
		context Event inv:
		`self.trigger=true implies not`
		`LogicalOperator.XOR and LogicalOperator.OR`

Control Flow	Control Flow Connector	A control flow connector is only allowed to connect not more than 2 of function, event and logical operator.
		`context Control Flow Connector inv:`
		`function + event + operator->sum() <=2`

Activity Partition	Organisation Unit	An organisation unit is assigned to one ore more functions.
		`context OrganisationUnit inv:`
		`self.Function->size() >= 1`
		An organisation unit is assigned to one ore more organisation roles.
		`context OrganisationUnit inv:`
		`self.OrganisationRole->size() >= 1`

Activity Partition	Organisation Role	An organisation role is assigned to one ore more functions.
		`context OrganisationRole inv:`
		`self.Function->size() >= 1`

Data Store Node	Information Object	An information object is assigned to one ore more functions.
		`context InformationObject inv:`
		`self.Function->size() >= 1`

Object Node	Deliverable	A deliverable is assigned to one ore more functions.
		`context Deliverable inv:`
		`self.Function->size() >= 1`

5 Example

We show the practical applicability of the UML 2 profile for EPCs with the example business process of an insurance company: *Processing of Automobile Insurance*

Claims. The process is modelled with EPCs (Fig. 3) as well as with the UML 2 Profile for EPCs (Fig. 4), subsequently we compare both.

Every new claim from an insured person is recorded and calculated by a financial expert. Depending on the calculated insurance sum the process is divided into a path for a minor or a major amount. For minor damages, the garage will be contacted for information about the damage. For major damages, the insurance history of the customer will be checked and the garage will be also contacted. After the examination of results, a decision will be made. If the decision is positive, the insurance company will pay for the damage, otherwise no bank transfer will be done.

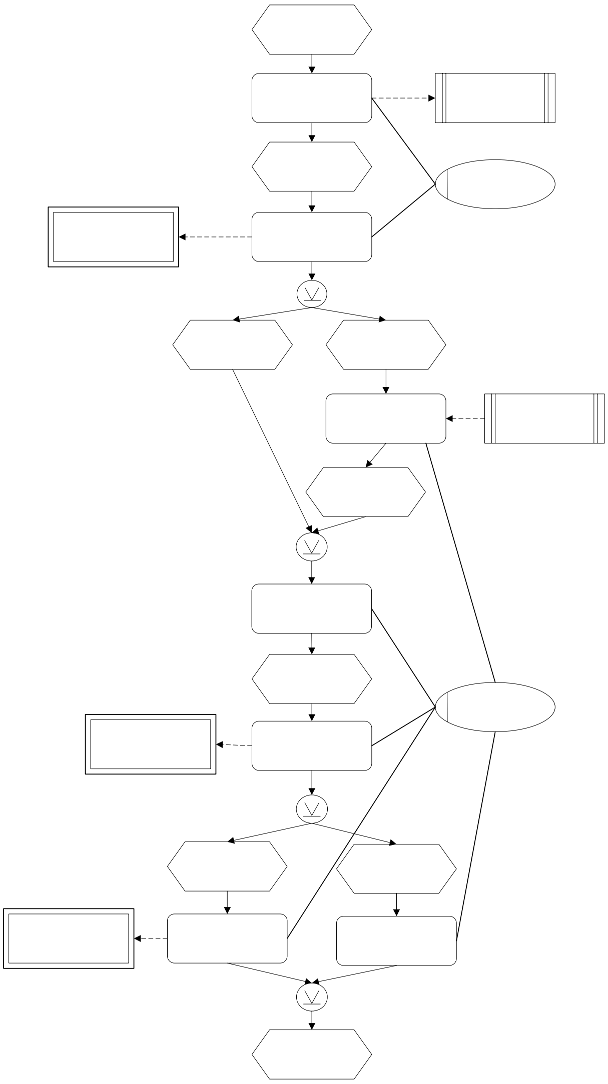


Fig. 3. Processing of Automobile Insurance Claims in EPC Notation

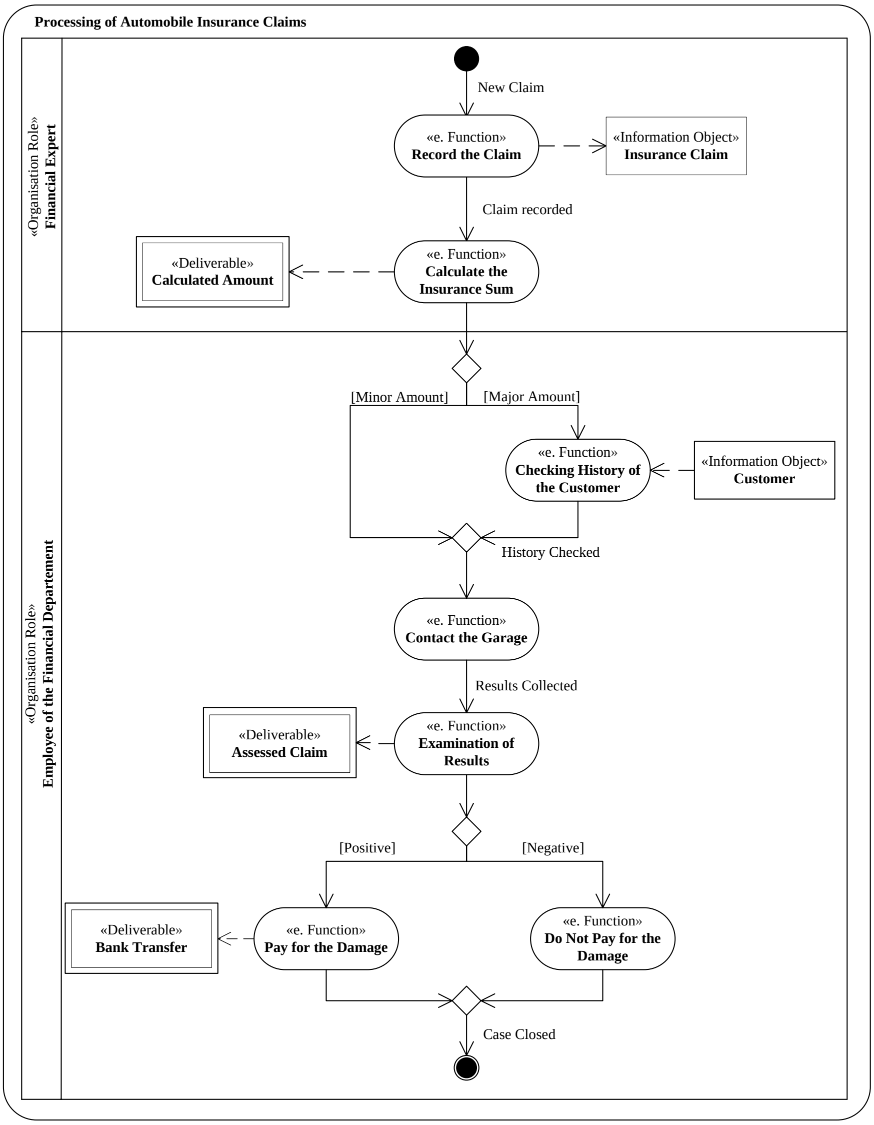

Processing of Automobile Insurance Claims
«Organisation Role» Financial Expert
«Organisation Role» Employee of the Financial Departement
New Claim
«e. Function» Record the Claim
«Information Object» Insurance Claim
Claim recorded
«Deliverable» Calculated Amount
«e. Function» Calculate the Insurance Sum
[Minor Amount]
[Major Amount]
«e. Function» Checking History of the Customer
«Information Object» Customer
History Checked
«e. Function» Contact the Garage
Results Collected
«Deliverable» Assessed Claim
«e. Function» Examination of Results
[Positive]
[Negative]
«Deliverable» Bank Transfer
«e. Function» Pay for the Damage
«e. Function» Do Not Pay for the Damage
Case Closed

Fig. 4. Processing of Automobile Insurance Claims in UML 2 Notation

Comparing the EPC process model in Fig. 3 with the UML2 profile for EPCs process model in Fig. 4, we can see that the latter is not running into complexity and very well readable. It appears almost more structured than the original EPC model because firstly, the organisational structure is at the boundary of the model. Secondly, the EPC events, which are sometimes creating additional complexity without any benefit, are in the background of the model. In both cases, the elements of the process model are reduced. Thus, the developed UML 2 profile is more structured than the original EPC.

6 Related Work

In the current literature, there are already some UML profiles for business process modelling available. The profiles focus on the sequential flow of the business process, but do not represent EPCs at all. Most existing profiles are based on UML 1.4., whereby the UML 2 profile for EPCs is based on UML 2.

The UML Profile for Business Modeling [6] is defined in the UML 1.4 specification and embodies the object-oriented approach for business engineering developed by Jacobson et al. [2]. The model lacks a detailed process flow. Johnston extended [6] with an activity diagram, called the Rational UML Profile for business modeling [3].

The UML 2 profile for Business Process Modelling in [4] captures business process characteristics from a wide angle like the customer, the process owner, goals, measures and products, but does not consider the detailed flow.

Activity Nets - a UML profile for modeling workflow and business processes [1] targets the modelling of business process architectures and concurrent processes.

The UML profile for Business Modelling in [9] is focusing on the integration of business processes into software development. The profile maps between business concepts and software artefacts and describes the process flow in a very a detailed way.

7 Conclusion

In this paper, we developed a UML 2 profile for EPCs targeting software developers to view EPC models in a familiar notation. The profile is realised as a 1-1 mapping between EPCs and UML 2 activity diagrams. Each element of the EPC meta-model was described with stereotypes. The profile was tested with an example process.

References

[1] Bochmann, G. v.: A UML profile for modelling workflow and business processes. http://beethoven.site.uottawa.ca/dsrg/PublicDocuments/Publications/Boch00d.pdf

[2] Jacobson, I., Ericson, M., Jacobson, A.: The Object Advantage – Business Process Reengineering with Object Technology. ACM Press, Addison-Wesely Publishing 1995.

[3] Johnston, S.: Rational UML Profil for Business Modeling. http://www-128.ibm.com/developerworks/rational/library/5167.html#author1

[4] List, B. and Korherr, B.: A UML 2 Profile for Business Process Modelling, Proceedings of the 1st International Workshop on Best Practices of UML at the 24th International Conference on Conceptual Modeling (ER 2005), Austria, Springer Verlag 2005.

[5] Object Management Group, Inc.: UML 2.0 OCL Specification http://www.omg.org

[6] Object Management Group, Inc.: UML 1.4 Specification http://www.omg.org/

[7] Object Management Group, Inc.: UML 2.0 Superstructure http://www.omg.org/

[8] Scheer, A.-W.: ARIS – Business Process Modeling. Springer Verlag 1999.

[9] Tyndale-Biscoe, S., Sims, O., Wood, B., Sluman C.: Business Modelling for Component Systems with UML. Proceedings of the Sixth International Enterprise Distributed Object Computing Conference (EDOC '02), IEEE Press 2002.