Transformation Algorithms of Knowledge Based UML Dynamic Models Generation

Ilona Veitaite(✉) and Audrius Lopata

Department of Informatics, Kaunas Faculty, Vilnius University,
Muitines g. 8, 44280 Kaunas, Lithuania
{Ilona.Veitaite,Audrius.Lopata}@knf.vu.lt

Abstract. The article represents knowledge based UML Dynamic models (Use Case, Activity, State Machine, Protocol State Machine, Sequence, Communication, Timing and Interaction Overview) generation process from Enterprise Model (EM), where every model generation is defined with transformation algorithm. The algorithms description is presented as Activity diagrams with depiction an explanation of essential steps.

Keywords: Enterprise modelling · Knowledge-based · IS engineering · UML · Transformation algorithms

1 Introduction

Business and IT alignment is a main concern in both fields: IT and business. There are a broad variety of models, methods and techniques i.e. Zachman framework, J. Henderson and N. Venkatraman business and IT strategic alignment model likewise numerous IS engineering frameworks. Majority of them introduces guidelines on the abstract theoretical degree only and practical realization solutions on engineering degree is nevertheless not sufficient [1, 3, 4].

The usage of formal structure called knowledge-based subsystem, which consists of Enterprise model and Enterprise meta-model, improves the quality and sufficiency of IS engineering process. The Enterprise meta-model is assumed to be the essential formal structure for domain knowledge gathering for the IS development targets. This meta-model regulates Enterprise model structure and Enterprise model accumulates knowledge that is required for whole IS development process and will be used in all stages of IS development life cycle [1, 5, 7, 8].

UML is one of the most extensive software specification standards. It is a universal IS modelling language used in a number of methodologies and enforced in well-known modelling tools [1]. The knowledge stored in Enterprise model can be applied in UML models dedicated to use in IS development life cycle stages [11, 12]. All these UML can be generated within transformation algorithms, when the necessary knowledge is gathered into knowledge repository, where it is already verified to assure automatically generated design models quality in knowledge collection into knowledge repository phase [13, 14].

W. Abramowicz (Ed.): BIS 2017 Workshops, LNBIP 303, pp. 59–68, 2017.
https://doi.org/10.1007/978-3-319-69023-0_6

2 Matching Between Enterprise Model Elements and UML Elements

Enterprise meta-model is formally determined enterprise model composition, which contained of a formalized enterprise model alongside with the general principles of control theory (Fig. 1). Enterprise model is the main source of the requisite knowledge of the specific problem domain for IS engineering and IS reengineering processes [3].

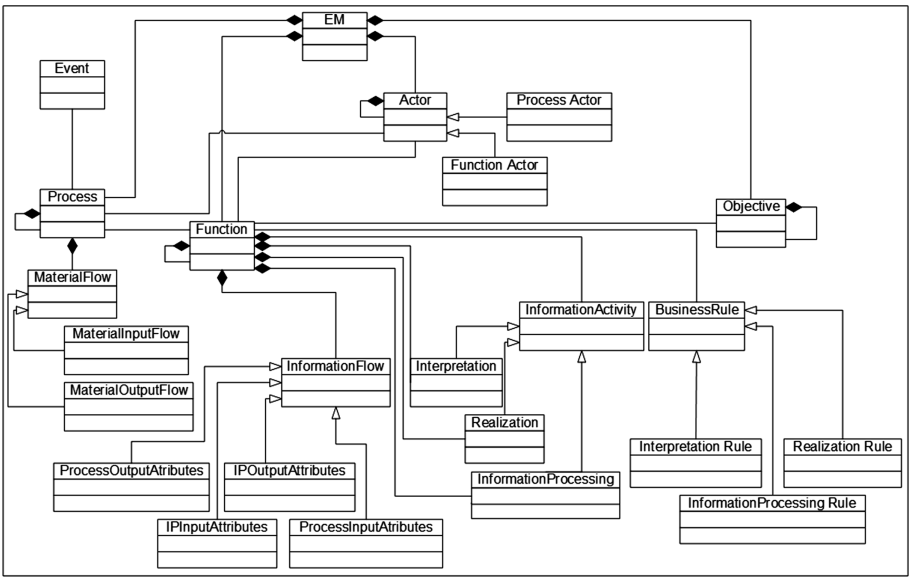

Fig. 1. Class diagram of Enterprise meta-model [3]

UML models can be generated from Enterprise model using transformation algorithms. Firstly, certain UML model must be identified for generation process, after this identification, the initial – main element of this UML model must be selected from Enterprise model. Secondly, all related elements must be selected according the initial element and all these related components must be linked between regarding constraints, necessary for UML model type identified earlier.

Information systems design methods specify the arrangement of systems engineering actions, i.e. how, in what order and what UML model to use in the IS development process and how to implement the process (Table 1). Many of them are based on diverse types of models describing differing aspects of the system properties. Sense of each model can be defined individually, but more important is the fact that each model is the projection of the system. An inexperienced specialist can use UML models inappropriately and the description of the system will possibly be contradictory, insufficient and contentious [1, 2, 6].

Table 1. Enterprise model process and function elements role variations in UML dynamic models [9, 10].

EM	UML model element	UML dynamic model	Description
Process/function	Use case	Use case model	A use case is a type of behavioural classifier that defines a unit of functionality achieved by actors or subjects to which the use case applies in combination with one or more actors
	Activity	Activity model	Describes a parameterized behaviour as correlative flow of actions
	Behavioural state machine	State machine model	Specifies individual behaviour of a part of designed system through limited state transitions
	Protocol state machine	Protocol state machine model	Expresses a usage protocol or a lifecycle of some classifier
	Message	Sequence model	Describes one specific kind of communication between lifelines of an interaction
	Frame	Communication model	Describes a unit of behaviour that concentrates on the appreciable exchange of information between connectable elements
	Frame	Interaction overview model	Describes a unit of behaviour that concentrates on the appreciable exchange of information between connectable elements

Information systems design methods specify the arrangement of systems engineering actions, i.e. how, in what order and what UML model to use in the IS development process and how to implement the process. Many of them are based on diverse types of models describing differing aspects of the system properties. Sense of each model can be defined individually, but more important is the fact that each model is the projection of the system. An inexperienced specialist can use UML models inappropriately and the description of the system will possibly be contradictory, insufficient and contentious [1, 2, 6].

This kind of system description is totally confusing, because most of the information in the models overlay and express the same things just in different approaches, as it is shown in the table (Table 1). Accordingly identification of exact UML model for generation process has high importance, because of regarding this selection depends generated element significance for system development process.

3 UML Dynamic Model Transformation Algorithms

UML Dynamic models represent the dynamic behaviour of the objects in a system, which can be described as a series of changes to the system over time [10].

3.1 UML Use Case Model Transformation Algorithm

UML Use Case model defines a collection of actions, called use cases that some system or systems, called subject, should or can operate in combination with one or more external users of the system, called actors, to contribute some appreciable and relevant outcomes to the actors or other users of the system or systems [9, 10].

In UML Use Case model generation from Enterprise model (Fig. 2) initial element is actor or subject, after generation of this element, follows selection of Enterprise model element: process or function and use case element is generated. After the generation of these two types of elements, they have to be linked to each other with some type of relationship: association, extension or inclusion, which is defined by Enterprise model element Business Rule. After all these elements are generated, there is update of actor or subject element and check are there more actor elements left in enterprise model.

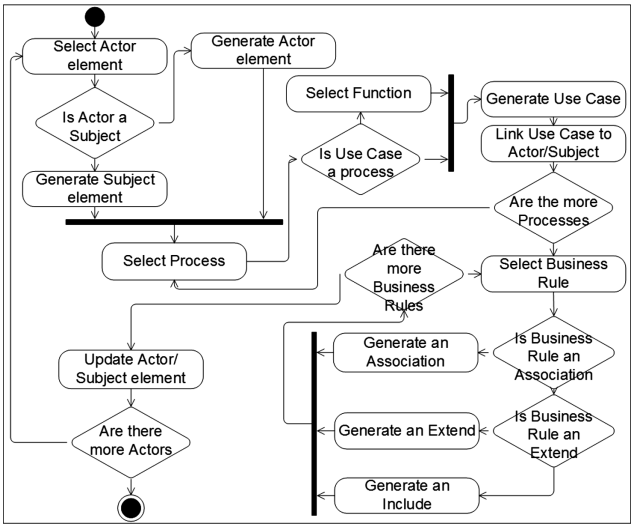

Fig. 2. UML use case model transformation algorithm

3.2 UML Activity Model Transformation Algorithm

UML Activity model describes sequence and conditions for coordinating lower-level behaviours, instead than which classifiers own those. These behaviours are generally called control flow and object flow models [9, 10].

In UML Activity model generation from Enterprise model (Fig. 3) initial element is partition, after generation of this element, follows selection of Enterprise model element: process or function and activity element is generated. Afterward the generation of these two types of elements, they have to be linked to each other. In activity models object nodes are generated from Enterprise model material or informational flow elements. And all these generated elements are connected through control nodes which are based on Enterprise model business rule elements. Later all these elements are generated, there is update of actor element and check are there more actor elements left in enterprise model.

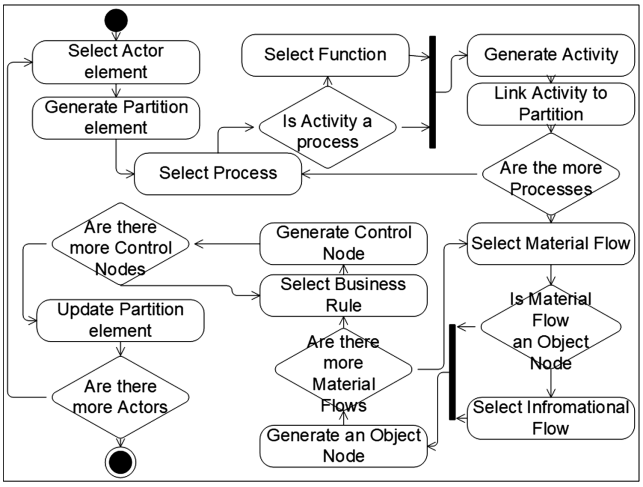

Fig. 3. UML activity model transformation algorithm

3.3 UML State Machine and Protocol State Machine Models Transformation Algorithms

UML State Machine models are used for modelling individual behaviour through definite state transitions. To express the behaviour of a part of the system, state machines can further be used to express the usage protocol of part of a system. These two types of state machines are committed to as behavioural state machines and protocol state machines. First type of State machine model describes individual behaviour of a part of designed system through limited state transitions [9, 10].

In UML Behavioural State Machine model generation from Enterprise model (Fig. 4) initial element is process or function, it means that from these enterprise model elements behavioural state machine element is generated. Subsequently this element generation second related element is simply state or composite state, which is generated from information flow. Furthermore, first two elements are linked to each other and also with pseudostate element, which is generated from business rule. After that there is update of the initials element and check are there more process elements left in enterprise model [9, 10].

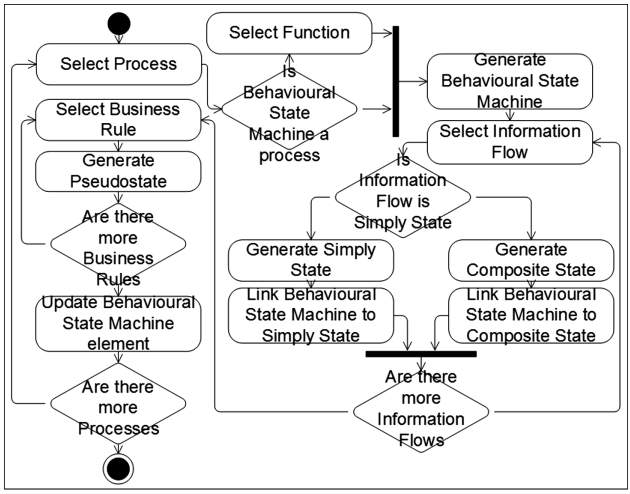

Fig. 4. UML state machine model transformation algorithm

And second type of State machines is UML Protocol State Machine model, which defines usage protocol or a lifecycle of some classifier [10].

In UML Protocol State Machine model generation from Enterprise model (Fig. 5) initials element is also process or function, from these enterprise model elements protocol state machine element is generated. After this element generation information flow is selected and Protocol state element generated. These two elements are linked to each other and also with protocol transition element, which is generated from business rule. Afterwards that there is update of the initial element and check are there more process elements left in enterprise model.

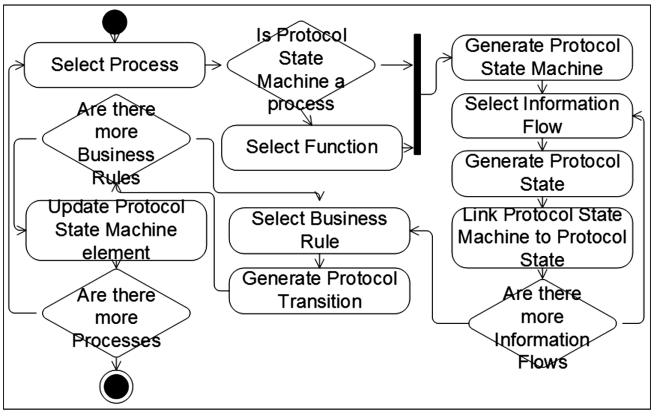

Fig. 5. UML protocol state machine model transformation algorithm

3.4 UML Sequence Model Transformation Algorithm

UML Sequence model is one of the interaction models group which concentrates on the message interchange between system participants called lifelines [9, 10].

In UML Sequence model generation from Enterprise model (Fig. 6) initial element is lifeline, after generation of this element, follows selection of Enterprise model element: process or function and message element is generated. Afterwards the generation of these two types of elements, they have to be linked to each other. In sequence models execution specification, combined fragment, interaction use, state invariant and destruction occurrence are generated from Enterprise model business rule elements. Later all these elements are generated, there is update of lifeline element and check are there more actor elements left in enterprise model.

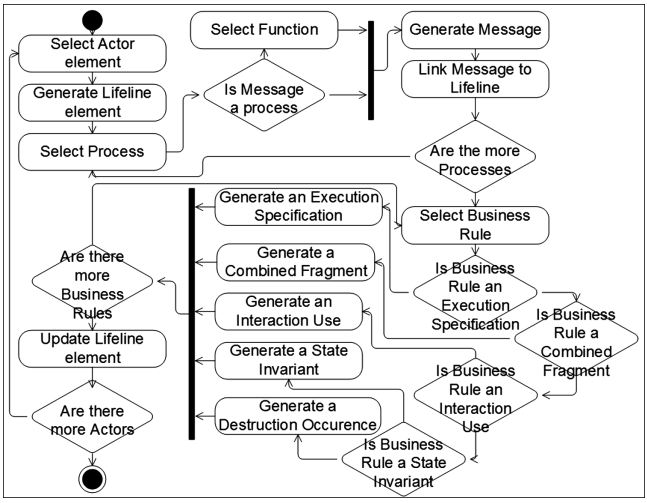

Fig. 6. UML sequence model transformation algorithm

3.5 UML Communication Model Transformation Algorithm

UML Communication Model is another one of the interaction models group, and it focuses on the interaction between participants called lifelines where the architecture of the internal structure and how this corresponds with the objects called message passing is central [9, 10].

In UML Communication model generation from Enterprise model (Fig. 7) initial element is lifeline, after generation of this element, follows selection of Enterprise model element: process or function and frame element is generated. Subsequently the generation of these two types of elements, they have to be linked to each other. In communication models message elements are generated from information flow element. Later all these elements are generated, there is update of lifeline element and check are there more actor elements left in enterprise model.

3.6 UML Timing Model Transformation Algorithm

UML Timing model is also one of the interaction models group and defines interactions when a primary purpose of the model is to reason about time. Timing models concentrate on conditions changing inside and between lifelines ahead a linear time axis [9, 10].

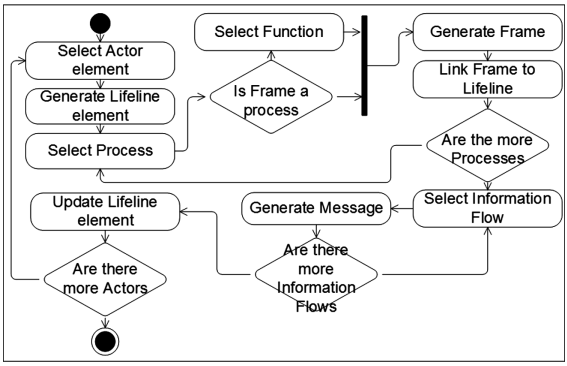

Fig. 7. UML communication model transformation algorithm

In UML Timing model generation from Enterprise model (Fig. 8) initial element is lifeline, after generation of this element, follows selection of Enterprise model element information flow and timeline or duration constraint element is generated. Subsequently the generation of these two types of elements, they have to be linked to each other. In timing models time constraint and destruction occurrence elements are generated from business rule element. Later all these elements are generated, there is update of lifeline element and check are there more actor elements left in enterprise model.

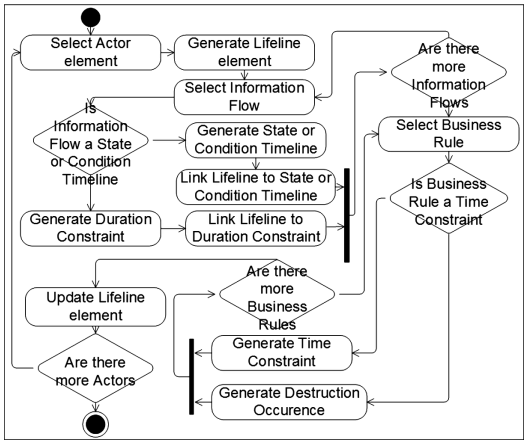

Fig. 8. UML timing model transformation algorithm

3.7 UML Interaction Overview Model Transformation Algorithm

UML Interaction Overview model is last of the interaction models group, which defines interactions through a variant of models in a way that stimulates overview of the control flow. Interaction overview models focus on the overview of the flow of control where the nodes are interactions or interaction uses. The participants like lifelines and objects like messages do not appear at this overview level [9, 10].

In UML Interaction model generation from Enterprise model (Fig. 9) initial element is process or function, it means that from these enterprise model elements frame element is generated. All other elements: duration constraint, time constraint, interaction use and control nodes, are related to the initial frame element and depends from enterprise business rule element.

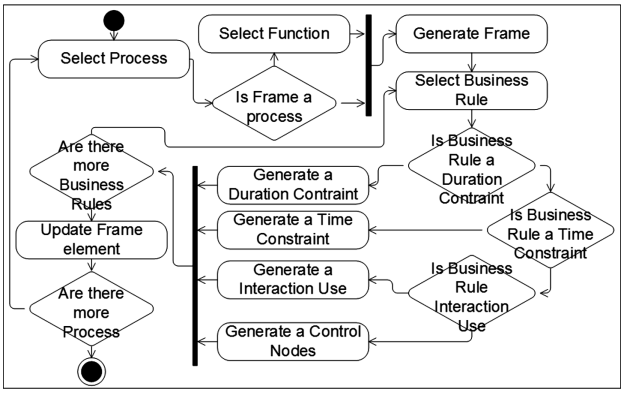

Fig. 9. UML interaction overview model transformation algorithm

4 Conclusions

In the first part of the article the matching of Enterprise model elements and UML Dynamic model elements is presented. The next part handles with detailed explanation of UML Dynamic models (Use Case, Activity, State Machine, Protocol State Machine, Sequence, Communication, Timing and Interaction Overview) transformation, that are necessary for knowledge-based IS engineering process are presented. The specified solution insures data coherence between specific design models in this manner providing more systematic IS engineering process.

The future works are: to create implementation of these transformation algorithms by knowledge-based tool's prototype, to include more detailed UML models elements application instruction into IS engineering process and to determine more validation preferences also present progressive enterprise model analysis aspects.

References

1. Butleris, R., Lopata, A., Ambraziunas, M., Veitaite, I., Masteika, S.: SysML and UML models usage in knowledge based MDA process. Elektronika ir elektrotechnika. **21**(2), 50–57 (2015). Print ISSN 1392-1215, Online ISSN 2029-5731
2. Chen, H.Y., Li, Ch., Tse, T.H.: Transformation of UML interaction diagrams in to contract specifications for object-oriented testing. Postprint of Article in Proceedings of the 2007 IEEE International Conference on Systems, Man, and Cybernetics (SMC 2007). IEEE Computer Society, Los Alamitos (2007)
3. Gudas, S., Lopata, A.: Meta-model based development of use case model for business function. Inf. Technol. Control **36**(3) [8] (2007). ISSN 1392 – 124X 2007
4. Henderson, J., Venkatraman, N.: Strategic alignment: leveraging information technology for transforming organizations. IBM Syst. J. **38**(2), 472–484 (1999)
5. IEEE Computer Society: Guide to the Software Engineering Body of Knowledge SWEBOK. Version 3.0. (2014). Paperback ISBN-13: 978-0-7695-5166-1
6. Nikiforova, O., Kozacenko, L., Ahilcenoka, D.: UML sequence diagram: transformation from the two-hemisphere model and layout. Appl. Comput. Syst. **14**, 31–41 (2013/14). doi:10.2478/acss-2013-0004
7. Lopata, A., Ambraziunas, M., Gudas, S.: Knowledge based MDA requirements specification and validation technique. Transform. Bus. Econ. **11**(1), 248–261 (2012)
8. Lopata, A., Ambraziunas, M., Gudas, S., Butleris, R.: The main principles of knowledge-based information systems engineering. Electron. Electr. Eng. **11**(1), 99–102 (2012). ISSN 2029-5731
9. OMG UML: Unified Modelling Language version 2.5. Unified Modelling (2017). http://www.omg.org/spec/UML/2.5
10. UML diagrams: The Unified Modeling Language (2017). http://www.uml-diagrams.org/
11. Lopata, A., Veitaite, I.: UML diagrams generation process by using knowledge-based subsystem. In: Abramowicz, W. (ed.) BIS 2013. LNBIP, vol. 160, pp. 53–60. Springer, Heidelberg (2013). doi:10.1007/978-3-642-41687-3_7
12. Veitaitė, I., Ambraziūnas, M., Lopata, A.: Enterprise model and ISO standards based information system's development process. In: Abramowicz, W., Kokkinaki, A. (eds.) BIS 2014. LNBIP, vol. 183, pp. 73–79. Springer, Cham (2014). doi:10.1007/978-3-319-11460-6_7
13. Veitaitė, I., Lopata, A.: Enterprise model, MOF and ISO standards based information system's development. XX tarpuniversitetinė magistrantų ir doktorantų konferencija. Informacinės technologijos 2015. Konferencijos pranešimų medžiaga. Vilniaus universitetas (2015). ISSN 2029-249X
14. Lopata, A., Veitaite, I., Žemaitytė, N.: Enterprise model based UML interaction overview model generation process. In: Abramowicz, W., Alt, R., Franczyk, B. (eds.) BIS 2016. LNBIP, vol. 263, pp. 69–78. Springer, Cham (2017). doi:10.1007/978-3-319-52464-1_7. ISBN 978-3-319-26762-3