Problem Domain Knowledge Driven Generation of UML Models

Ilona Veitaite(✉) and Audrius Lopata

Kaunas Faculty, Institute of Applied Informatics, Vilnius University, Kaunas, Lithuania
{Ilona.Veitaite,Audrius.Lopata}@knf.vu.lt

Abstract. The main scope of the article is to present how the quality of stored problem domain information in Enterprise model (EM) is significant and important in Unified Modelling Language (UML) models generation process from Enterprise model. The generation process is explained by top-level transformation algorithm, which is presented in details and depicted by using algorithm's step by step description. The importance of information quality and fullness is represented by the example of Business Rule element's, which is stored in Enterprise model, significance for different UML models.

Keywords: UML · Unified Modelling Language · Enterprise model Transformation algorithm · Knowledge-based IS engineering

1 Introduction

There have been and still are pretty many efforts for the analysis of UML models generation from different knowledge-based models, merging and uniting different modelling languages, frameworks and patterns and even generation from natural language specifications [2–4, 9].

The Enterprise meta-model (EMM) is considered to be the significant conceptual structure for problem domain knowledge acquisition for the aims of IS development. This meta-model handles Enterprise model structure, and therefore, Enterprise model stores knowledge that is needed for whole IS development process and can be used in all IS development life cycle phases [5–8].

In recent years UML models are having a rising regard from. It is a very daring objective for analysis of UML models since the knowledge about an enterprise system is distributed within several model approaches. UML models are preserved to decrease the confusion of the problem with the increase of enterprise turnovers. By operating UML models, knowledge from Enterprise model can be efficiently represented and can be used in all phases of IS development life cycle [10, 11, 13].

2 Generation of UML Models Using Enterprise Meta-model

Enterprise meta-model is formally defined enterprise model structure, which consists of a formalized Enterprise model in line with the general principles of control theory (Fig. 1). Enterprise model is the main source of the necessary knowledge of the particular problem domain for IS engineering and IS re-engineering processes [6, 7].

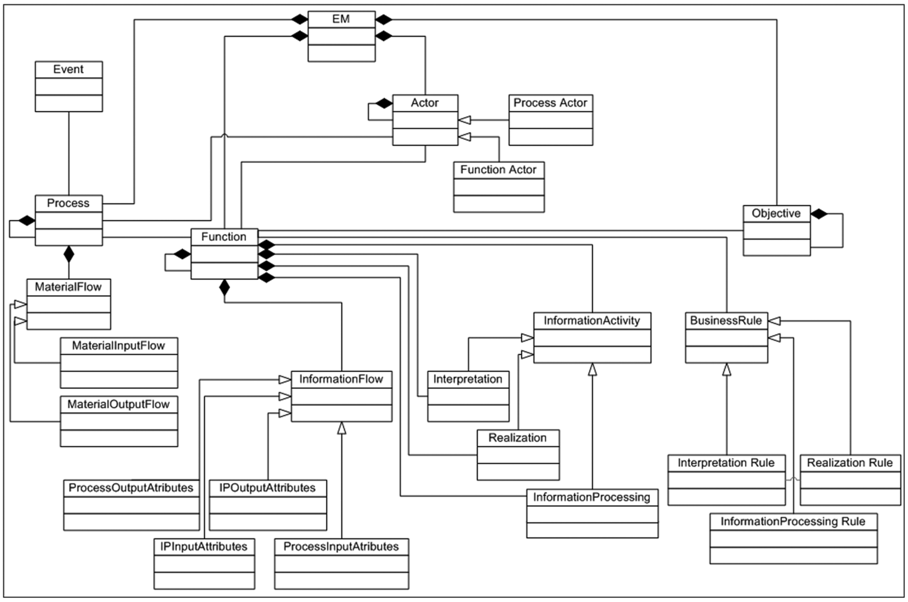

Fig. 1. Enterprise meta-model class diagram [6, 7]

All UML models: static and dynamic can be generated using Enterprise model transformation algorithms. Particular UML model must be selected for generation process, after this selection, the initial – starting element of this particular UML model must be identified from Enterprise model. Therefore, all related elements must be identified according the initial element and all these related elements must be linked regarding constraints i.e. business rules, obligatory for particular UML model type which was selected in the beginning [1, 12].

This kind of system definition is quite tangled, because most of the information from the Enterprise model overlays in the UML models and expresses the same matters just in different approaches, as it is explained in the next chapter. Therefore identification of particular UML model for generation process has high significance, because of regarding this selection relies generated element value for system development process [1, 16].

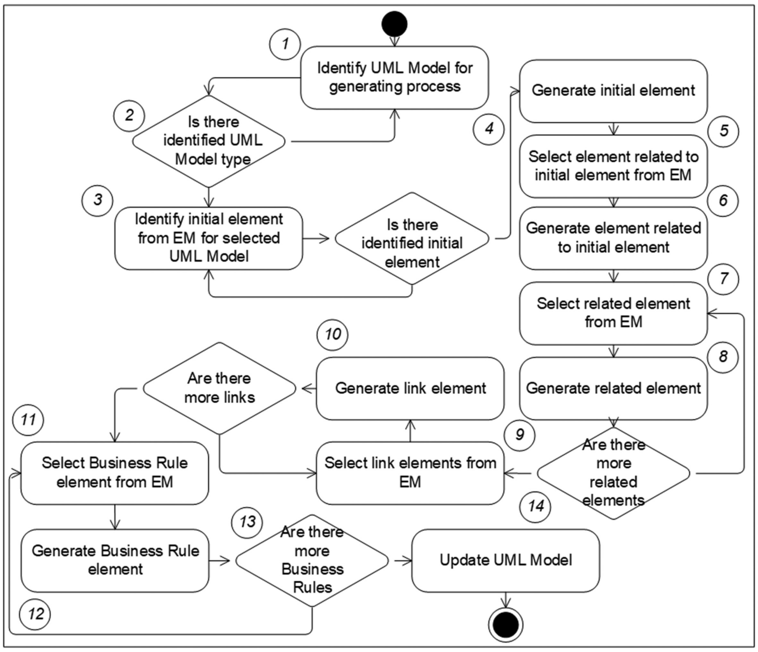
1
Identify UML Model for generating process
2
Is there identified UML Model type
3
Identify initial element from EM for selected UML Model
Is there identified initial element
4
Generate initial element
5
Select element related to initial element from EM
6
Generate element related to initial element
7
Select related element from EM
8
Generate related element
Are there more related elements
9
Select link elements from EM
10
Generate link element
Are there more links
11
Select Business Rule element from EM
Generate Business Rule element
12
13
Are there more Business Rules
Update UML Model
14

Fig. 2. The top level algorithm of UML models generation using normalised EMM

The algorithm of UML model generation using Enterprise model is presented in the Fig. 2:

- Step 1: Particular UML model for generation from Enterprise model process is identified and selected.
- Step 2: If the particular UML model for generation from Enterprise model process is selected then algorithm process is continued, else the particular UML model for generation from Enterprise model process must be selected.
- Step 3: First element from Enterprise model is selected for UML model, identified previously, generation process.
- Step 4: If the selected Enterprise model element is initial UML model element, then initial element is generated, else the other Enterprise model element must be selected (the selected element must be initial element).
- Step 5: The element related to the initial element is selected from Enterprise model.
- Step 6: The element related to the initial element is generated as UML model element.
- Step 7: The element related to the previous element is selected from Enterprise model.
- Step 8: The element related to the previous element is generated as UML model element.

- Step 9: If there are more related elements, then they are selected from Enterprise model and generated as UML model elements one by one, else the link element is selected from Enterprise model.
- Step 10: The link element is generated as UML model element.
- Step 11: If there are more links, then they are selected from Enterprise model and generated as UML model elements one by one, else the Business Rule element is selected from Enterprise model.
- Step 12: The Business Rule element is generated as UML model element.
- Step 13: If there are more Business Rules, then they are selected from Enterprise model and generated as UML model elements one by one, else the generated UML model is updated with all elements, links and constraints.
- Step 14: Generation process is finished.

3 Business Rule Element's Variations

Design methods of information systems refers the settlement of systems engineering actions, i.e. in what manner, how, and which UML model to use in the information system development process and how to fulfil the process. Many of them are based on different types of models depicting varying aspects of the system properties. Significance of every UML model element can be defined particularly, but more important is the fact that every model is the projection of the system (Fig. 3). Business Rule element from Enterprise model can indicate different view of the system in different UML model.

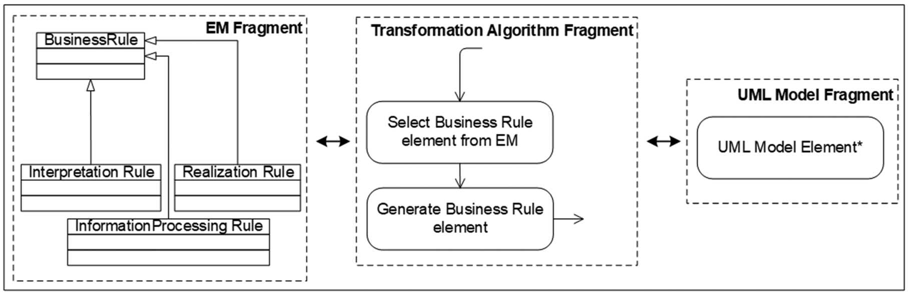
EM Fragment
BusinessRule
Interpretation Rule
Realization Rule
InformationProcessing Rule
Transformation Algorithm Fragment
Select Business Rule element from EM
Generate Business Rule element
UML Model Fragment
UML Model Element*

Fig. 3. Business Rule element in EM and in UML model. The transformation algorithm's 11 and 12 steps

Below there are presented generated UML model elements after transformation algorithm's 11 and 12 steps: Business Rule element generation from Enterprise model in UML dynamic part models.

In case of UML Use Case models, which describe a series of actions that some system or systems should or can implement in contribution with one or more external users of the system [15], from EM Business Rule elements can be generated three UML Use Case elements (Table 1): Extend, Include, Association [14, 15].

Table 1. UML Use Case model elements [14, 15]

EM element	UML Use Case model element	Description
Business Rule	Extend	Extend is a directed relationship that specifies how and when the behaviour defined in usually supplementary (optional) extending use case can be inserted into the behaviour defined in the extended use case
	Include	Use case include is a directed relationship between two use cases which is used to show that behaviour of the included use case is inserted into the behaviour of the including use case
	Association	Each use case represents a unit of useful functionality that subjects provide to actors. An association between an actor and a use case indicates that the actor and the use case somehow interact or communicate with each other

Table 2. UML Activity model elements [14, 15]

EM element	UML Activity model element	Description
Business Rules	Control Nodes	Used to coordinate the flows between other nodes. It includes: initial, flow final, activity final, decision, merge, fork, join

In case of UML Activity model, which shows flow of control or object flow with emphasis on the sequence and conditions of the flow [15], from EM Business Rule elements can be generated all types of UML Activity Control Nodes elements (Table 2).

In case of UML State machine model, which is used for modelling discrete behaviour through finite state transitions [15], from EM Business Rule elements can be generated one UML State machine model element: Pseudostate (Table 3), and also state machines can be used to express the usage protocol of part of a system as UML Protocol State machine, and in this case, from EM Business Rule elements can be generated one UML Protocol State machine model element: Protocol Transition (Table 4).

Table 3. UML State Machine model elements [14, 15]

EM element	UML State Machine model element	Description
Business Rule	Pseudostate	An abstract node that encompasses different types of transient vertices in the state machine graph

Table 4. UML Protocol State Machine model elements [14, 15]

EM element	UML State Machine model element	Description
Business Rule	Protocol Transition	Used for the protocol state machines which specifies a legal transition for an operation

In case of UML Sequence model, which focuses on the message interchange between objects (lifelines) [15], from EM Business Rule elements can be generated five UML Sequence model elements (Table 5): Execution Specification, Combined Fragment, Interaction Use, State Invariant and/or Destruction Occurrence.

Table 5. UML Sequence model elements [14, 15]

EM element	UML Sequence Model element	Description
Business Rules	Execution Specification	Represents a period in the participant's lifetime
	Combined Fragment	Defines a combination (expression) of interaction fragments. A combined fragment is defined by an interaction operator and corresponding interaction operands
	Interaction Use	Allows to use (or call) another interaction
	State Invariant	Represents a runtime constraint on the participants of the interaction
	Destruction Occurrence	Represents the destruction of the instance described by the lifeline

In case of UML Timing model, which shows interactions when a primary scope of the model is to reason about time [15], from EM Business Rule elements can be generated three UML Time model elements (Table 6): Duration Constraint, Time Constraint, Destruction Occurrence.

Table 6. UML Timing model elements [14, 15]

EM element	UML Timing Model element	Description
Business Rules	Duration constraint	Refers to a duration interval. The duration interval is duration used to determine whether the constraint is satisfied
	Time Constraint	Refers to a time interval. The time interval is time expression used to determine whether the constraint is satisfied
	Destruction Occurrence	Represents the destruction of the instance described by the lifeline

In case of UML Interaction Overview model, which identifies interactions through a variant of activity models in a way that sustains overview of the control flow [15], from EM Business Rule elements can be generated five Interaction Overview model elements (Table 7): Duration Constraint, Time Constraint, Interaction Use, Control Nodes.

Table 7. UML Interaction Overview model element [14, 15]

EM element	UML Interaction Overview model element	Description
Business Rules	Duration Constraint	Refers to a duration interval. The duration interval is duration used to determine whether the constraint is satisfied
	Time Constraint	Refers to a time interval. The time interval is time expression used to determine whether the constraint is satisfied
	Interaction Use	Allows to use (or call) another interaction
	Control Nodes	Used to coordinate the flows between other nodes. It includes: initial, flow final, activity final, decision, merge, fork, join

This wide group of elements can be generated because UML Interaction Overview model coordinates elements from activity and interaction models [15]:

- from the activity model: initial node, flow final node, activity final node, decision node, merge node, fork node, join node;
- from the interaction models: interaction, interaction use, duration constraint, time constraint.

All these UML models elements descriptions shows, how much knowledge about problem domain is stored in Enterprise model, how much information behind the EM Business Rule element is saved and also, how important that this stored information's quality and fullness should match all the different UML models approaches, so that suitable UML model could be generated.

4 Conclusions

The first part of the article handles with defining Enterprise model concept and with detailed explanation of Unified Modelling Language model generation from Enterprise model transformation algorithm, which is depicted by steps.

In the next part there are presented the Business Rule element's variations in different UML models, which were generated from Enterprise model, and how the quality of the business problem information, stored in knowledge-based Enterprise model makes the impact on these variations.

The described Business Rule element's example shows that if the high quality information stored in Enterprise model, then it is enough for UML models generating process. There is possible to confirm, that each element of each UML dynamic model can be generated from the Enterprise model using transformation algorithms only if high quality of the information in Enterprise model is ensured.

References

1. Butleris, R., Lopata, A., Ambraziunas, M., Veitaitė, 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. Čyras, V.: Intelektualios sistemos. Vilnius: Vilniaus universitetas, Matematikos ir informatikos fakultetas, Programų sistemų katedra, 116 p. (2017)
3. Evensen, K.D., Weiss, K.A.: A comparison and evaluation of real-time software systems modeling languages. In: AIAA Infotech@Aerospace 2010, 20–22 April 2010, Atlanta, Georgia (2010)
4. IEEE Computer Society: Guide to the Software Engineering Body of Knowledge SWEBOK. Version 3.0. (2014). Paperback ISBN-13: 978-0-7695-5166-1
5. Gudas, S.: Enterprise knowledge modelling: domains and aspects. Technol. Econ. Dev. Econ. Baltic J. Sustain., 281–293 (2009)
6. Gudas, S.: Architecture of knowledge-based enterprise management systems: a control view. In: Proceedings of the 13th world Multiconference on Systemics, Cybernetics and Informatics (WMSCI 2009), vol. 3, pp. 161–266, Orlando, Florida, USA, 10–13 July 2009. ISBN -10: 1-9934272-61-2 (Volume III). ISBN -13: 978-1-9934272-61-9 (Volume III)
7. Gudas S.: Informacijos sistemų inžinerijos teorijos pagrindai. Vilniaus universiteto leidykla (2012). ISBN 978-609-459-075-7
8. Gudas, S., Lopata A.: Meta-model based development of use case model for business function. Inf. Technol. Control **36**(3) (2007). ISSN 1392 – 124X 2007
9. Küster, T., Lützenberger, M., Heßler, A., Hirsch, B.: Integrating process modelling into multi-agent system engineering. multi-agent and grid systems (2012). https://www.researchgate.net/publication/220535349_Integrating_process_modelling_into_multi-agent_system_engineering
10. Lopata, A., Veitaite, I., Gudas, S., Butleris, R.: CASE tool component – knowledge-based subsystem. UML diagrams generation process. Transformations Bus. Econ. **13**(2B) (32B), 676–696 (2014). ISSN: 1648 - 4460
11. Lopata, A., Ambraziūnas, M., Gudas, S., Butleris, R.: The main principles of knowledge-based information systems engineering. Electron. Electr. Eng. **11**(1)(25), 99–102 (2012). ISSN 2029-5731
12. Lopata, A., Ambraziūnas, M., Gudas, S.: Knowledge based MDA requirements specification and validation technique. Transformations Bus. Econ. **11**(1)(25), 248–261 (2012)
13. Lopata, A., Ambraziūnas, M., Gudas, S.: Knowledge-based MDA requirements specification and validation technique. Transformations Bus. Econ. **11**(1(25)), 248–260 (2012). ISSN 1648-4460
14. OMG UML: Unified Modelling Language version 2.5. Unified Modelling (2018). http://www.omg.org/spec/UML/2.5

15. UML diagrams (2018). https://www.uml-diagrams.org/
16. 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). https://doi.org/10.1007/978-3-319-11460-6_7