Problem Domain Example of Knowledge-Based Enterprise Model Usage for Different UML Behavioral Models Generation

Ilona Veitaite[1(✉)] and Audrius Lopata[2]

[1] Kaunas Faculty, Institute Social Sciences and Applied Informatics, Vilnius University, Muitines Street 8, 44280 Kaunas, Lithuania
ilona.veitaite@knf.vu.lt

[2] Faculty of Informatics, Kaunas University of Technology, Student Street 50, 51368 Kaunas, Lithuania
audrius.lopata@ktu.lt

Abstract. The main purpose of this paper is to represent how knowledge-based Enterprise Model (EM) as problem domain data storage may be used in Information Systems (IS) engineering process. Enterprise Meta-Model (EMM) presented more than two decades ago justifies EM structure. EM stores problem domain data gathered by analyst and this EM structure can be used for project models creation in IS design phase. Unified Modeling Language (UML) models are one of the possible models, which can be generated from EM. These models can be generated through transformation algorithms. To present possibility of UML models generation from EM particular problem domain example is defined in this paper. Presented example demonstrates that data stored in EM is enough for different UML models generation and this paper presents that different UML behavioral models can be generated from EM and can illustrate same problem domain from different perspectives.

Keywords: IS engineering · Transformation algorithm · Enterprise modeling · Knowledge-based · UML

1 Introduction

Nowadays IS engineering process is still challenging for all IT professionals: analysts, designers, developers. They have same goal but all are responsible for different phase of IS engineering process [1, 5, 7]. Beginning of this process is most important, because problem domain analysis is performed and final result will depend from quality of gathered data. Duration of IS engineering process and number of errors during it may also impact the final result. So it is very important how gathered data of problem domain will be used, where it will be stored and what project models standards will be chosen [1, 4].

There are a wide number of models to store problem domain data, also there are a lot of modeling standards and notations. UML models are widely applied in IS engineering process and are used as by IT professionals in IS design phase as by not this field professionals for better understanding of final IS result [4, 5, 7]. UML latest version 2.5 [6, 8]. UML models may be designed one by one according to collected

W. Abramowicz et al. (Eds.): BIS 2021 Workshops, LNBIP 444, pp. 45–55, 2022.
https://doi.org/10.1007/978-3-031-04216-4_5

data by analyst, they can be created by using certain data storages assigned for modeling and design phase and also they can be generated from particular EM [5, 7]. EM used in this analysis is created more than two decades ago and the main goal of this article is to present its efficiency for UML models generation process. For this purpose, there are created transformation algorithms defined in previous researches. By using these algorithms UML models generation from EM is possible [2, 3, 10, 12]. Car Rental Company example and generated UML models of this example, which defines problem domain information from different perspectives presented in this research depict how knowledge stored in EM can be used for UML models generation.

2 Knowledge-Based EM Definition

EMM is formally defined EM structure, which consists of a formalized EM in line with the general principles of control theory. EM is the main source of the necessary knowledge of the particular business domain for IS engineering and IS re-engineering processes [2, 3, 10, 12].

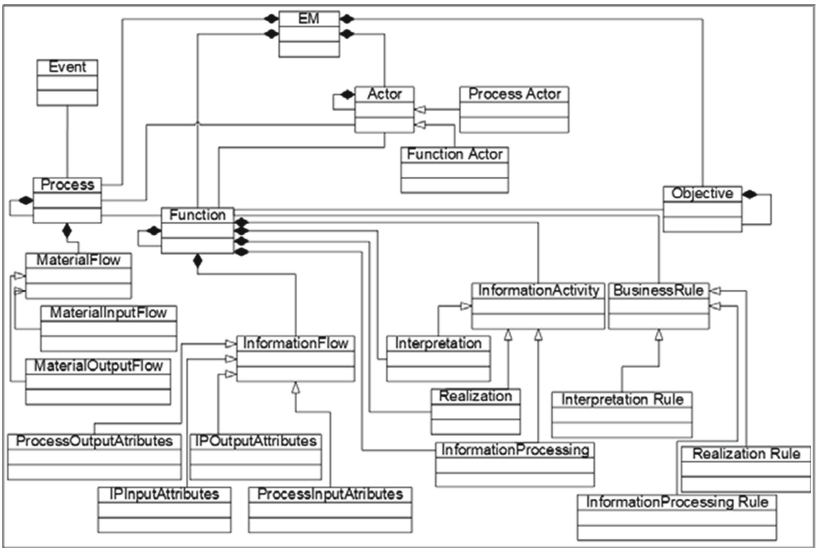

Fig. 1. Class diagram of Enterprise Meta-Model [2, 3, 9]

EM consists of twenty-three classes (Fig. 1). Essential classes are Process, Function and Actor. Class Process, Function, Actor and Objective can have an internal hierarchical structure. These relationships is presented as aggregation relationship. Class Process is linked with the class MaterialFlow as aggregation relationship. Class MaterialFlow is linked with the classes MaterialInputFlow and MaterialOutputFlow as generalization relationship. Class Process is linked with Classes Function, Actor and

Event as association relationship. Class Function is linked with classes InformationFlow, InformationActivity, Interpretation, InformationProcessing and Realization as aggregation relationship. These relationships define the internal composition of the Class Function. Class InformationFlow is linked with ProcessOutputAtributes, ProcessInputAtributes, IPInputAttributes and IPOutputAttributs as generalization relationship. Class InformationActivity is linked with Interpretation, InformationProcessing and Realization as generalization relationship. Class Function linked with classes Actor, Objective and BusinessRule as association relationship. Class BusinessRule is linked with Interpretation Rule, Realization Rule, InformationProcessing Rule as generalization relationship. Class Actor is linked with Function Actor and Process Actor as generalization relationship [2, 3, 10, 12].

During IS engineering analysis phase analyst gather all necessary information and stores it in EM for further IS engineering design phase, when system project models are created. There are a lot of standards and different notations of project model and UML models are most commonly used by IT professionals. With the EM usage, IS design phase project models can be generated.

3 UML Models Transformation Algorithm

Each of structural or behavioral UML models can be generated through transformation algorithm and each of models has separate transformation algorithm [9–11]. These transformation algorithms are presented in previous researches. Main focus of researches is dedicated for generation behavioral or dynamic UML models, because they are more complex and variable [12–14]. To have better understanding of transformation algorithm itself, top level transformation algorithm of UML models generation from EM process is described step by step [9–14]:

- Step 1: Particular UML model for generation from EM process is identified and selected.
- Step 2: If the particular UML model for generation from EM process is selected then algorithm process is continued, else the particular UML model for generation from EM process must be selected.
- Step 3: First element from EM is selected for UML model, identified previously, generation process.
- Step 4: If the selected EM element is initial UML model element, then initial element is generated, else the other EM element must be selected (the selected element must be initial element).
- Step 5: The element related to the initial element is selected from EM.
- 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 EM.
- 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 EM and generated as UML model elements one by one, else the link element is selected from EM.
- Step 10: The link element is generated as UML model element.
- Step 11: If there are more links, then they are selected from EM and generated as UML model elements one by one, else the Business Rule element is selected from EM.
- 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 EM 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.

Usually main problem domain knowledge necessary for IS engineering process is stored in particular EM elements (commonly mandatory for most used UML models) [12–14]:

- Actor: in actor element can be stored information related with process or function executor. Actor element is responsible of information related with the process or function participant, it can be person, group of persons, subject such as an IS, subsystem, module and etc.
- Process or Function: in process or function elements can be stored all information related with any user, entity, object, subject and its behavior. Process or function element is responsible of information related with any operation, activity, status change, movement which is implemented by any actor, entity, participant and etc.
- InformationFlow: in Information Flow element can be stored diverse information flow types, such as Information input and output attributes or/and process input and output attributes. Information Flow element is responsible of information related with each element input and output attributes, details which make impact on other elements, their state or status.
- MaterialFlow: in Material Flow element can be stored two types of material information Material input and Material output. Material Flow element is responsible of information related with any material flows of the described process or function.
- BusinessRule: in Business Rule element can be stored different rules such as interpretation, realization or/and information processing. Business rule element is responsible of information about how different elements in IS design phase are related; what restrictions and restraints are applied to these elements.

4 Generated UML Models of Car Rental Company

To present sufficiency of knowledge-based EM example of particular problem domain – Car Rental Company – is analyzed. Car Rental process may be defined as car rental process management system that manages rental process with participant of two users: client and manager. Client may enquire for a rental car, if after client verification,

documents check and car availability check all requirements are satisfied, client may rent a car; after the usage client returns car and pays for services, and manager controls this process: calculates fees, receives payment and maintains returned car.

4.1 UML Use Case Model

UML Use Case Model can be generated from EM, because all necessary elements for UML Use Case Model are stored in it. Generation process follows the steps of transformation algorithm. Connections between EM and UML Use Case Model elements are described in Table 1.

Table 1. EM and UML use case model elements of car rental company [6, 8, 9, 11, 12]

EM element -> UML element	Car Rental Company example
Actor -> Actor	There are two participants: Client, who wants to rent a car, enquires for it and pays for the service and Manager, who controls car renting process, verifies the client, calculates payment and etc
Process, Function -> Use Case	There are eleven Use Cases, part of them are performed by Client, part – by Manager. There are also common for both participant Use Cases, only functionality is different
BusinessRule -> Association, Include, Extend	Use Cases are related with particular actors. Also there is defined which Use Cases are necessary to perform and which aren't

Table 1 defines EM and UML Use Case Model elements and defines their meaning in Car Rental Company example. Generated UML Use Case Model is presented in Fig. 2.

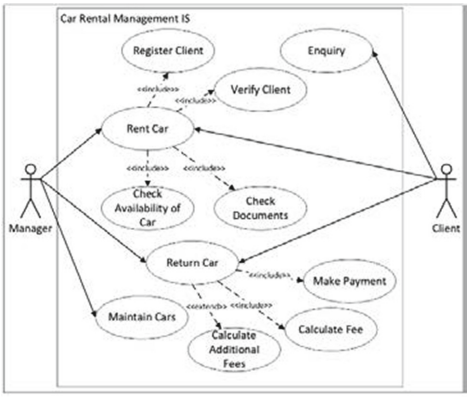

Fig. 2. Generated UML use case model of car rental company

Figure 2 presents UML Use Case Model generated from EM through transformation algorithm. It defines how two participants – Actors perform their activities – Use Cases linked by Association relationship, which of the must be performed – Include relationship and which not – Extend relationship.

4.2 UML Activity Models

UML Activity Models (only for two processes (rent and return car) in this research. Note: number of UML Activity Models of this example can be higher) can be generated from EM, because all necessary elements for UML Activity Models are stored in it. Generation process follows the steps of transformation algorithm. Connections between EM and UML Activity Models elements are described in Table 2.

Table 2. EM and UML Activity model elements of Car Rental Company [6, 8, 9, 12]

EM element -> UML element	Car Rental Company example
Actor -> Actor, Swimlane	In Fig. 3 UML Activity Model of renting car, there are two participants: Manager and Client. In Fig. 4 UML Activity Model there is only one participant: Manager
Process, Function -> Activity	Figure 3 presents activities performed by two participants: Client and Manager and Fig. 4 only from the perspective of the Manager
MaterialFlow, InformationFlow -> Object Flows	In both UML Activity Models all activities are related through Object Flows
BusinessRule -> Control Nodes: Initial, Join, Decision, Final	Both UML Activity Models start from initial nodes and end with final nodes. Both models have Decision nodes and Join nodes

Table 2 defines EM and UML Activity Models elements of two diagrams and defines their meaning in Car Rental Company example. Generated UML Activity Models are presented in Fig. 3 and Fig. 4.

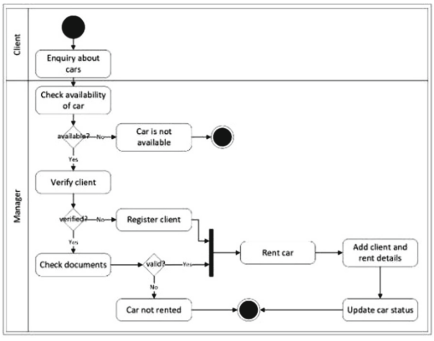

Fig. 3. Generated UML activity model of rent car process

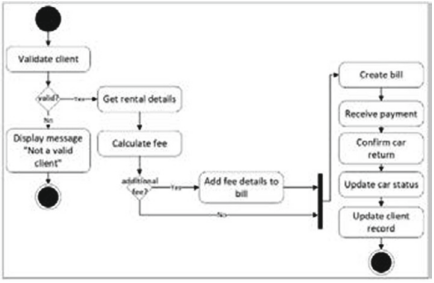

Fig. 4. Generated UML activity model of return car process

Figure 3 presents UML Activity Model of Rent Car Process generated from EM through transformation algorithm. It defines how two participants – Swimlanes: Manager and Client perform the activities related with Car renting process; there is presented beginning of the process, flow of the activities, decisions made during the process and possible endings of the process.

Figure 4 presents UML Activity Model of Return Car Process generated from EM through transformation algorithm. It defines how one participant – Manager performs the activities related with Car returning process; there is presented beginning of the process, flow of the activities, decisions made during the process and possible endings of the process.

4.3 UML State Models

UML State Models (only for two objects (car and manager) in this research. Note: number of UML State Models of this example can be higher) can be generated from EM, because all necessary elements for UML State Models are stored in it. Generation process follows the steps of transformation algorithm. Connections between EM and UML State Models elements are described in Table 3.

Table 3. EM and UML state model elements of car rental company [6, 8, 9, 12]

EM element -> UML element	Car rental company example
Process, Function -> Behavioral state machine	In Fig. 5 UML State Model there are defined states of the Car and in Fig. 6 UML State Model there are defined states of the Manager
InformationFlow -> Composite state	In both UML State Model information about each object/machine is provided as their state

Table 3 defines EM and UML State Models elements of two diagrams and defines their meaning in Car Rental Company example. Generated UML State Models are presented in Fig. 5 and Fig. 6.

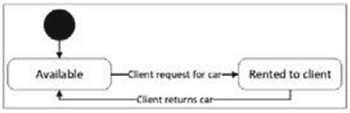

Fig. 5. Generated UML state model of car

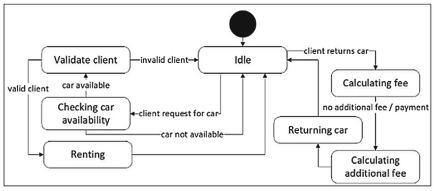

Fig. 6. Generated UML state model of manager

Figure 5 presents UML State Model of Rent Car Process generated from EM through transformation algorithm. It defines different states of the Car.

Figure 6 presents UML State Model of Return Car Process generated from EM through trans-formation algorithm. It defines different states of the Manager.

4.4 UML Communication Models

UML Communication Models (only for two processes (rent and return car) in this research. Note: number of UML Communication models of this example can be higher) can be generated from EM, because all necessary elements for UML Communication Models are stored in it. Generation process follows the steps of transformation algorithm. Connections between EM and UML Communication Models elements are described in Table 4.

Table 4. EM and UML communication model elements of car rental company [6, 8, 9, 12]

EM element -> UML element	Car rental company example
Actor -> Lifeline	In Fig. 7 UML Communication Model there are four Lifelines: ClientRecord, Manager, Transaction and Car and they are directly related with Car renting process. In Fig. 8 UML Communication Model there are five Lifelines, Invoice is new one and they are directly related with Car returning process
Process, Function -> Frame	Figure 7 presents Car renting process and Fig. 8 present Car returning process
InformationFlow -> Message	In both UML Communication Models messages between all Lifelines are presented
BusinessRule -> Sequence Expression	In both UML Communication Models Sequence Expressions define sequence of the messages

Table 4 defines EM and UML Communication Models elements of two diagrams and defines their meaning in Car Rental Company example. Generated UML Communication Models are presented in Fig. 7 and Fig. 8.

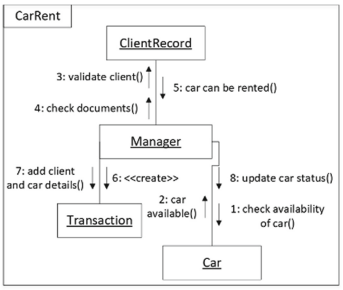

Fig. 7. Generated UML communication model of renting car

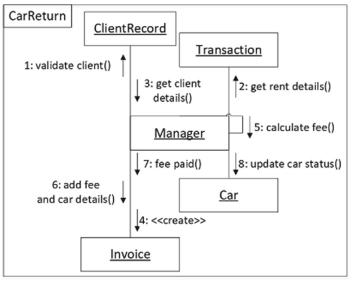

Fig. 8. Generated UML communication model of returning car

Figure 7 presents UML Communication Model of Rent Car Process generated from EM through transformation algorithm. It defines how four Lifelines: Manager, ClientRecord, Transaction and Car interacts during Car renting process.

Figure 8 presents UML Communication Model of Return Car Process generated from EM through transformation algorithm. It defines how five Lifelines: Manager, ClientRecord, Transaction, Car and Invoice interacts during Car returning process.

4.5 UML Sequence Models

UML Sequence Models (only for two processes (rent and return car) in this research. Note: number of UML Sequence Models of this example can be higher) can be generated from EM, because all necessary elements for UML Sequence Models are stored in it. Generation process follows the steps of transformation algorithm. Connections between EM and UML Sequence Models elements are described in Table 5.

Table 5. EM and UML sequence model elements of car rental company [6, 8, 9, 12]

EM element -> UML element	Car rental company example
Actor -> Lifeline	In Fig. 9 UML Sequence Model there are four Lifelines: ClientRecord, Manager, Transaction and Car and they are directly related with Car renting process. In Fig. 10 UML Sequence Model there are five Lifelines, Invoice is new one and they are directly related with Car returning process
Process, Function -> Message	In both UML Sequence Models messages between all Lifelines are presented
BusinessRule -> Execution Specification, Occurrence Specification	In both UML Sequence Models Execution Specifications define durations of the executions and occurrences of the messages

Table 5 defines EM and UML Communication Models elements of two diagrams and defines their meaning in Car Rental Company example. Generated UML Communication Models are presented in Fig. 9 and Fig. 10.

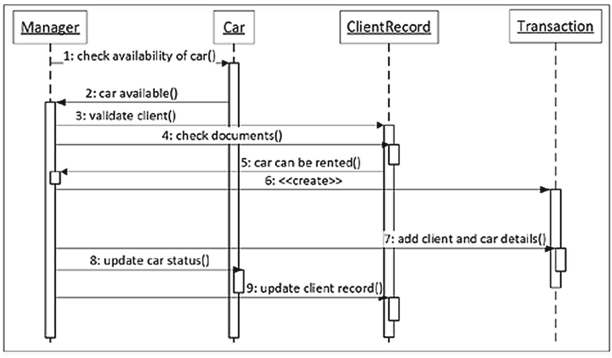

Fig. 9. Generated UML sequence model of renting car

Figure 9 presents UML Sequence Model of Rent Car Process generated from EM through transformation algorithm. It defines how four Lifelines: Manager, ClientRecord, Transaction and Car interacts during Car renting process, presents the sequence of Messages and duration of Executions between the Messages.

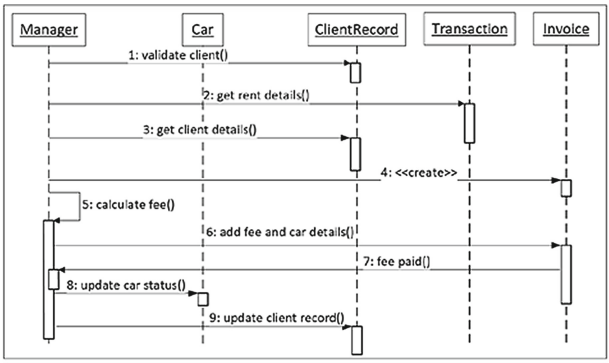

Fig. 10. Generated UML sequence model of returning car

Figure 10 presents UML Sequence Model of Return Car Process generated from EM through transformation algorithm. It defines how five Lifelines: Manager, ClientRecord, Transaction, Car and Invoice interacts during Car returning process, presents the sequence of Messages and duration of Executions between the Messages.

5 Conclusions

The first section of the article defines the Enterprise Model structure by presenting class diagram of EMM and listing all its elements also explaining their necessity in IS engineering process. Gathered problem domain data may be stored in EM and used to further IS development process.

The next section presents top level UML models transformation algorithm from EM depicted step by step. As UML is frequently used in IS design process, UML models may have great impact of the success of this process. In this section there are also defined commonly mandatory EM elements for most used UML models.

In final section description of particular problem domain is presented. In this research example of Car Rental Company is analyzed for the purpose to define that same problem domain may be presented from different perspectives with the help of UML models. All problem domain knowledge is stored in EM by analyst, and all this verified and validated information is used for different UML models generation by using transformation algorithms. UML Use Case, Activity, State, Communication and Sequence models are generated from EM. This list of generated models is not final, it is possible to generate all other UML models, depending on necessity to depict IS from different perspectives. These generated models confirm the sufficiency of EM in UML models generation process. Most important condition is that data stored in EM must be

acknowledged by analyst, because this procedure ensures lower number of possible errors and allows to avoid increased IS engineering process duration.

References

1. Dunkel, J., Bruns, R.: Model-driven architecture for mobile applications. In: Abramowicz, W. (ed.) BIS 2007. LNCS, vol. 4439, pp. 464–477. Springer, Heidelberg (2007). https://doi.org/10.1007/978-3-540-72035-5_36
2. Gudas, S.: Architecture of knowledge-based enterprise management systems: a control view. In: Proceedings of the 13th world multiconference on systemics, cybernetics and informatics (WMSCI2009),) July 10–13, Orlando, Florida, USA, vol. III, pp.161–266 (2009). ISBN -10: 1-9934272-61-2 (Volume III). ISBN-13: 978-1-9934272-61-9
3. Gudas S.: Informacijos sistemų inžinerijos teorijos pagrindai/ Fundamentals of Information Systems Engineering Theory. (Lithuanian)Vilnius University (2012). ISBN 978–609–459–075–7
4. Jacobson, I., Rumbaugh, J., Booch, G.: Unified Modeling Language User Guide, p. 0321267974. Addison-Wesley Professional, The Second Edition (2005)
5. Jenney, J.: Modern Methods of Systems Engineering: With an Introduction to Pattern and Model Based Methods (2010). ISBN-13:978-1463777357
6. OMG UML: Unified Modeling Language version 2.5.1. Unified Modelling (2021). https://www.omg.org/spec/UML/About-UML/
7. Sajja, P.S., Akerkar, R.: Knowledge-based systems for development. Adv. Knowl. Based Syst. Model, Appl. Res. **1** (2010)
8. UML Diagrams: UML diagrams characteristic (2021). www.uml-diagrams.org
9. Veitaite, I., Lopata, A.: Transformation algorithms of knowledge based UML dynamic models generation. In: Abramowicz, W. (ed.) BIS 2017. LNBIP, vol. 303, pp. 59–68. Springer, Cham (2017). https://doi.org/10.1007/978-3-319-69023-0_6
10. Veitaite, I., Lopata, A.: Problem domain knowledge driven generation of UML models. In: Damaševičius, R., Vasiljevienė, G. (eds.) ICIST 2018. CCIS, vol. 920, pp. 178–186. Springer, Cham (2018). https://doi.org/10.1007/978-3-319-99972-2_14
11. Abramowicz, W., Corchuelo, R. (eds.): BIS 2019. LNBIP, vol. 373. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-36691-9
12. Veitaite, I., Lopata, A.: Knowledge-based transformation algorithms of UML dynamic models generation from enterprise model. In: Dzemyda, G., Bernatavičienė, J., Kacprzyk, J. (eds.) Data Science: New Issues, Challenges and Applications. SCI, vol. 869, pp. 43–59. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-39250-5_3
13. Veitaitė, I., Lopata, A.: Knowledge-based UML activity model transformation algorithm. Information society and university studies 2020. In: Proceedings of the Information Society and University Studies, 2020, Kaunas, Lithuania, April 23, 2020. Audrius, L., Vilma, S., Tomas, K., Ilona, V., Marcin, W.A: CEUR Workshop Proceedings. ISSN 1613-0073. pp. 114–120 (2020). (CEUR Workshop Proceedings, ISSN 1613–0073; vol. 2698)
14. Veitaite, I., Lopata, A.: Knowledge-based generation of the UML dynamic models from the enterprise model illustrated by the ticket buying process example. In: Lopata, A., Butkienė, R., Gudonienė, D., Sukackė, V. (eds.) ICIST 2020. CCIS, vol. 1283, pp. 26–38. Springer, Cham (2020). https://doi.org/10.1007/978-3-030-59506-7_3