Topological Functioning Model for Software Development within MDA (Survey)

Arturs Solomencevs
Department of Applied Computer Science, Riga Technical University, Riga, Latvia

Keywords: Topological Functioning Model, Software Development, Formal Problem Domain Model, Model Driven Architecture.

Abstract: The approach called Topological Functioning Modeling for Model Driven Architecture (TFM4MDA) uses Topological Functioning Model (TFM) as a formal holistic problem domain model. The approach is revolutionary, because it brings formalism to the earliest stages of software development – the analysis of problem domain, and provides formal transformations to UML design models. A copious amount of effort has been put into the development of TFM4MDA. Furthermore, TFM has not always been used in software development. This paper represents a literature survey of 69 articles about TFM and its application. The goal of this work is to trace the research of TFM and TFM4MDA approach, to throw light on the results of the research, and to reveal some weaker areas of it. The goal is successfully achieved and the conclusions are made.

1 INTRODUCTION

In (Osis, 2004), it is stated that object orientation has become the dominate approach to the analysis and design of computerized systems. The Unified Modeling Language (UML) defines the industry-standard modeling notation for object-oriented software development. There are two fundamental aspects to modeling: analysis, which defines what the application has to do with the problem domain to fit the customer's requirements, and design, which defines how the application will be built.

The problem domain is the part of the world in which the software is required to bring about some effect desired by the customer, as it is described in (Osis, 2003b). The application domain is the software we build and the computer system that executes it. During the analysis and the design, both domains are modeled.

In paper (Osis, 2003a), it is asserted that modeling a problem domain brings essential advantages into software development. A proper problem domain model provides a powerful language for expressing requirements for the system. A precise problem domain model gives a precise architectural (application domain) model. Author of (Osis, 2001 b) claims that the stability of software architecture depends on the adequacy of problem domain model. In (Alksnis et al., 2005), it is proven that it is beneficial to model a problem domain formally. A formal model can be transformed into another model if the transformation rules are defined. For instance, a model which is a product of system analysis can be transformed into a model of system design. Also, formal design implies that the correctness of operation of the entire system is mathematically proven.

Unfortunately, as it is stated in (Osis, 2001e), UML and its application have some relevant drawbacks: 1) Use case driven object-oriented methods give low priority to problem domain modelling. Analyzing starts with application domain. In this case, problem domain is considered to be a "black box". 2) Even when use cases are applied for business modeling, relationship between a business system and a planned computerized one remain informal as well as identification of business and system use cases themselves (Asnina, 2006). 3) Diagrams (models) are constructed without a formal basis (Osis, 2001e). 4) UML goes without mathematics (Asnina and Osis, 2002).

Author of (Osis, 2003a) claims that the improvement of the results of object-oriented system analysis and modeling lies in using methods which are oriented to deal with problem domain, because the formalism must be involved on the very early

Solomencevs, A.
Topological Functioning Model for Software Development within MDA (Survey).
In *Proceedings of the 11th International Conference on Evaluation of Novel Software Approaches to Software Engineering (ENASE 2016)*, pages 315-326
ISBN: 978-989-758-189-2

stage of software development. Until 2003, a more or less formal way to map the knowledge about the problem domain into software development process did not exist.

Janis Osis from Riga Technical University invented Topological Functioning Model (TFM). Its theoretical fundamentals are published in (Osis, 1969). TFM is a formal model which describes the functioning of a system. TFM has a solid mathematical base. It is represented in a form of a topological space (X, Θ), where X is a finite set of functional features of the system under consideration, and Θ is topology that satisfies axioms of topological structures and is represented in a form of a directed graph. The TFM's functional features describe the system's physical or biological characteristics that are relevant for the normal functioning of the system. The TFM's topology consists of cause-effect relations between functional features. Cause-effect relation exists between two functional features, if appearance of one functional feature is caused by appearance of the other without participation of any intermediate functional feature. Cause-effect relations form causal chains. Causal chains must form at least one functioning cycle within TFM. All the cycles and subcycles should be carefully analyzed in order to completely identify existing functionality of the system.

The idea of applying TFM in object-oriented software engineering was published in (Osis, 2001b), and has been developed since then. It is the first milestone in the development of TFM. Researchers who made contributions to the TFM approach are: J. Osis (the leader), E. Asnina, U. Donins, A. Slihte, G. Alksnis, J. Silins and others. Vast amount of work has been put into the research of TFM and its application. The purpose of this paper is to review the approach and its development during the passage of time. This will throw light on the results of the research, and will reveal weaker areas which may need to be worked with. To achieve the goal, 69 papers were studied.

2 THE BEGINNING OF RESEARCH ON TFM

In papers (Osis, 2001a) and (Osis, 2001e), the development of TFM's theory is reviewed. In 1964, associate professor J. Osis at Riga Technical Universty begins the development of theoretical basis for diagnostics of complex systems. He also formed a scientific group. Since 1966, J Osis's PhD students have published oriented graph models in their works. The final theoretical fundamentals of TFM were published in (Osis, 1969).

In the beginning, TFM was not meant to be used in software development. By studying papers (Osis, 1972), (Gelfandbain et al., 1990) and (Osis et al., 1991) it becomes clear that topological modeling can be used for diagnostics of complex systems. TFM can also be applied in technical and medical diagnostics, in image recognition and in expert systems (Osis, 1991); in modeling of biological systems (Osis and Beghi, 1997); and in business process modeling and simulation (Osis et al., 1997). In paper (Ivasiuta and Osis, 1999), methods for comparing software design methodologies are reviewed.

Paper (Osis et al., 1996) is the first article about object-oriented approach that concerns TFM. In the paper, object-oriented modeling and simulation are discussed.

In paper (Osis, 1997), the development of object-oriented methods for hybrid system analysis and design is described.

The relation between domain modeling and architectural design is discussed in paper (Osis, 2001 b). Precise domain model gives precise architectural model. Stability of information system depends on the structure of problem domain. Thus, full knowledge about structure helps to minimize the risk of losing the stability.

Paper (Osis, 2001c) represents a survey of object-oriented approach. It was found that historically the idea that the world could be viewed either in terms of objects or processes comes from ancient philosophy and cognitive science and was an ancient Greek invention. In the seventeenth century Descartes declared that humans naturally apply an object-oriented view of the world. Analysis and software development methods are reviewed in the paper. Lack of formal basis for object-oriented modeling is discussed. The tendency of giving low priority to problem domain analysis in use case driven methods is disputed. The way to improve results of object-oriented system analysis and modelling lies in using formal methods which are oriented to deal with problem domain. In paper, topological modeling is mentioned as an approach for problem domain driven analysis. TFM supports abstraction and decomposition – two fundamental ways of dealing with system's complexity. Thus, it is possible to model complex systems with TFM.

In paper (Osis, 2001d), the development of object-oriented analysis is reviewed. It is stated that TFM's theoretical fundamentals give the opportunity

to get a strictly formalized model (TFM) from knowledge about a system. The obtained software architecture is adequate to problem domain.

The drawbacks of UML and its application are discussed in (Osis, 2001e). TFM is considered to overcome the mentioned disadvantages.

In papers (Alksnis and Osis, 2001) and (Alksnis and Osis, 2002), the research about how category theory can be applied in software development is carried out. The structure of topological space and TFM is very similar to the definition of category.

Paper (Osis and Silins, 2002) represents the analysis of modeling languages and methods which can be applied in development of embedded systems (real-time systems in particular). The paper can be considered to be an introduction to applying TFM in the development of embedded systems. However, TFM is not mentioned in the article.

In paper (Asnina and Osis, 2002), UML formalization possibilities using mechanisms of both universal arrow logic and topological modeling are described. Both approaches have strict mathematical basis – category theory. Formalization possibilities of the universal arrow logic and topological modeling are compared. TFM has an advantage over arrow logic, i.e., the operation of isolating the system from its topological space is formally defined – closure operation (Osis, 1969).

Paper (Nahimova and Osis, 2002) reviews the methods of modeling the mechatronic systems. TFM is mentioned as an approach that has advantage in modeling complex systems.

In paper (Osis, 2003a), the approach of using topological modeling for development of mechatronic and embedded systems is introduced. Paper (Osis, 2003b) introduces topological modeling for software engineering (TFM4SE) approach. It is stated that only a proper problem domain model provides a powerful language for expressing requirements to the system (Osis, 2003a), (Osis, 2003b). The construction of conceptual class diagram from TFM is defined. At this point, this model transformation is rather primitive, and is developed in future research. The joining operation of two TFMs is represented (Osis, 2003b).

3 TFM4MDA

Model Driven Architecture (MDA) is an approach to system development, which increases the power of models in this work. The purpose of MDA is to separate the views and concerns. MDA has three viewpoints and their corresponding models: a computation independent model (CIM) contains knowledge about the problem domain and the requirements for software system; platform independent model (PIM) focuses on the operation of a system while hiding the details necessary for a particular platform; and platform specific model (PSM) (Miller and Mukerji, 2003). Model transformation forms a key part of MDA. To get the software source code we need to go by the path CIM → PIM → PSM → source code.

Paper (Osis, 2004) introduces topological modeling for MDA. In the framework of MDA, TFM is used as a formal CIM (computation independent model). It is the first milestone in the development of TFM4SE. TFM for MDA is considered to be a subfield of TFM4SE.

The research about application of formal methods in development of embedded systems continues in (Alksnis et al., 2005). Topological modeling is included in these methods.

The idea of connecting MDA with software synthesis (code generation by applying artificial intelligence) is published in (Birgelis and Osis, 2005).

Papers (Osis, 2006a) and (Osis, 2006b) introduce novelties to TFM and MDA. TFM has systematic approach for checking its completeness and non-controversy (use case model, in its turn, does not have such an approach). TFM topological characteristics – *connectedness*, *closure*, *neighborhood* and *continuous mapping* – form mathematical background of TFM. TFM functional characteristics – *inputs*, *outputs*, *cycle structure* and *cause-effect relations* – form system theoretical background of the model. Thanks to these characteristics, TFM captures two aspects of the system – structure and dynamics. Modified MDA software development life cycle with a formal CIM – TFM – is introduced. In this cycle, feedback from the code deployment is not directed to the analysis phase like in the traditional life cycle, but to the TFM. Finally, the first metamodeling architecture is published. In this architecture, TFM is on M0 layer and UML Class diagram is on M1 layer.

In paper (Osis, 2006c), new metamodeling architecture is defined. Author emphasizes the importance of formalism in models, since formalized transformation can be automated.

A formal method of TFM constructing from a system's verbal description is given in (Asnina and Osis, 2006). This is an important milestone in the development of TFM4SE. A form for expressing TFM's functional features is defined:

<action>-ing the < result>

[to, into, in, by, of, from] a(n) <object>
e.g., Receiving the book from a reader.

The set of functional features should be written down in the following form:

<action>-ing a(n) <object>
e.g., Registering a reader.

Paper (Asnina, 2006) introduces the mapping of functional requirements (to planned information system) onto TFM's functional features. Mapping of requirements makes it possible to validate them in conformance to the business logic of the real world system as early as possible. During the mapping, TFM or (and) the list of requirements may need to be modified. The modified TFM represents the problem domain which is supported by the planned information system. Between requirements and a TFM the following mappings types exist: *One to One*; *Many to One*; *One to Many*; *One to Zero*; *Zero to One*; *Zero to Many*. Mapping types are explained in the article's text. For example, the existence of *Zero to Many* mapping may indicate that some requirement(s) is (are) missing. The paper also describes the algorithm for getting a use case diagram from TFM. By mapping requirements onto TFM and by creating use cases from TFM, it is possible to get an application domain model that conforms to problem domain knowledge.

In papers (Osis et al., 2007a) and (Osis et al., 2007b) TFM for MDA approach is called *TFMfMDA*. Papers introduce the explicit separation between problem domain and application domain. This is an important milestone in the development of TFM4SE. It is emphasized that functionality determines the structure of the planned system. TFM holistically represents complete functionality of the system. In papers, the mapping of requirements onto TFM and creation of use cases from TFM are developed further. The transformation "TFM → Graph of domain objects → Conceptual class diagram" is defined. Elements *precondition*, *the responsible entity* and *subordination* ("in" is inner, "ex" is external) are added to the expressing form of functional feature. Example:

Creating of a reader account, {unregistered person}, librarian, in.

Also, papers describe the requirements for a software tool for TFMfMDA automation. Thus, a new field of research – the development of tool support for the approach – is opened.

Papers (Osis et al., 2007c) and (Osis and Asnina, 2008a) introduce more formal way for describing a functional feature – a 5-tuple:

$<A, R, O, PrCond, E>$,

where A is an object action, R is a result of this action, O is an object(s) that receives the result or that is used in this action, *PrCond* is a set preconditions or atomic business rules (optional parameter), and E is an entity responsible for performing actions. The fields – mapping of requirements onto TFM; use case and conceptual class diagram creation from TFM – find further development in the papers. Also, a new transformation is defined "TFM → UML Activity diagram". Later, this transformation will be considered to be imprecise, and will be improved. The MOF-based metamodel of TFMfMDA is introduced. Also, the newest metamodeling architecture for TFMfMDA is published. This is a very important milestone in the development of TFM4SE. The architecture is not included because of space limits.

In papers (Asnina and Osis, 2008) and (Asnina et al., 2008) analysis and modeling of multifractal system properties in object-oriented software development is described. However, the papers do not specify the relation of this field with topological modeling.

Articles (Osis et al., 2008 a) and (Osis et al., 2008 b) give more detailed requirements for tool support for TFMfMDA approach.

In paper (Osis and Asnina, 2008b), new attributes are added to the tuple which describes a functional feature of TFM:

$<A, R, O, PrCond, PostCond, E, S>$,

where *PostCond* is a set of post-conditions (an optional element), and S is functional feature's belonging (subordination) to system's functionality (inner or external). Mapping of requirements onto TFM is developed further – the description of mapping type "*One to Zero*" is specified more precisely. Terms "source TFM" and "target TFM" are defined. Source TFM is a product of analyzing the problem domain. Target TFM is an output of mapping of requirements onto the functional features of TFM. Mapping ensures that target TFM is in compliance with source TFM.

4 TopUML

The intention to create an extension of UML named "Topological Unified Modeling Language" is expressed in paper (Osis, 2003a). The main motive is to introduce mathematical formalism into UML

diagrams. On the other hand, further refinement of TFM is required. It should be noted as the first milestone in the development of TopUML. The aim of "TopUML" research is to create a new version of UML that can be applied in a formal way which allows clearly tracing cause-and-effect relationships between the problem and solution domains. Solution domain is a system (e.g., business system) which is supported by the planned information system. To achieve this goal a new language is developed – Topological Unified Modeling Language (TopUML) – and its supporting software development method – TopUML modeling. TopUML is considered to be a research subfield of TFM4SE. In this subsection, articles which relate to TopUML are reviewed. The chief researcher is U. Donins, and his supervisor is J. Osis.

The main goal of papers (Osis and Donins, 2009a), (Donins and Osis, 2009), (Osis and Donins, 2009b), (Donins, 2010) and (Osis and Donins, 2010a) is to bring formalism into UML Class diagram, and to introduce a formal approach for creating a UML Class diagram that conforms to the "target" TFM. The creation of conceptual class diagram from TFM, which is described in (Osis and Asnina, 2008a), is disputed, because the topology between the classes is not retained. Therefore, topological relations between classes are defined. Topological relations are not included in OMG UML standard. Thus, topological (TopUML) class diagram, which contains topological relations, is defined. Transformation "TFM → Graph of domain objects → Topological class diagram" is introduced in the papers. New attributes are added to the functional feature tuple:

$$<A, R, O, PrCond, PostCond, E, \underline{Cl}, \underline{Op}>,$$

where *Cl* is a class and *Op* is an operation. These new attributes are used in the creation of graph of domain objects from TFM. Also, *S* (subordination) attribute of the tuple is not used in these papers.

Paper (Osis and Donins, 2010b) describes OMG MOF-compatible metamodel of Topological class diagram. This metamodel is needed to create the UML profile for Topological class diagram, which is proposed in the paper. This is a very important milestone in the development of TFM4SE.

In paper (Donins et al., 2011), refinement process of Topological class diagram is presented. The goal of this process is to lower the abstraction level of the initial Topological class diagram which is obtained from the TFM. Formal and informal guidelines for the refinement are given.

Paper (Donins and Osis, 2011) introduces transformation "TFM → Graph of domain objects → UML Sequence diagram". Separate sequence diagram is obtained for each system goal and requirement. Also, a case study of applying topological modelling approach to information system development is shown.

The research in (Donins, 2012a) introduces new element to TFM – logical relations. The analysis of logical relations helps to identify topological relations in TFM. Also, it helps to verify the consistency of TFM. Logical relations are needed for transformation of TFM into other models. In paper, an improved transformation "TFM → UML Activity diagram" is defined. This transformation, thanks to logical relations, gives a more precise output than the one defined in (Osis et al., 2007c). Formal definitions with tuples are given for the following TFM elements: preconditions, postconditions, topological and logical relationships. The tuples and their descriptions are not included because of limited space of this survey.

Transformation "TFM → UML Communication diagram → Topological class diagram" is defined in (Donins et al., 2012a) and (Donins et al., 2012b). So, in the transformation from TFM to Topological class diagram, communication diagram or graph of domain objects can be used as intermediate model (see article (Osis and Donins, 2009a)). Another transformation – "TFM → set of State diagrams" – is defined. One State diagram is obtained for each class. State diagrams give opportunity to analyze event-driven software systems. New attributes are added to the functional feature tuple:

$$<A, R, O, PrCond, PostCond, E, S, Cl, Op, \underline{St}, \underline{Es}>,$$

where *St* – new state of object *O* after performing action *A* (optional, needed for transformation to State diagrams), *Es* - indicates if execution of action *A* could be automated.

Paper (Osis et al., 2014) summarizes the research results of TopUML and U. Donins's PhD thesis (Donins, 2012b). TopUML allows creating the most popular UML design models from TFM. Some models are obtained successively, i.e., using an intermediate model. For example, at first one needs to obtain a Communication diagram from TFM, and only then – a Topological class diagram. All transitions between TFM and TopUML diagrams can be found in paper (Osis et al., 2014), in Section IV.B. Also, the paper's research explores traceability of modeling artifacts - from functioning properties and functional requirements of the problem domain to the software design and development artifacts.

The trace links are set by the mentioned transitions between diagrams.

TFM approach concentrates on formal analysis of problem domain. The output of this analysis is a "target" TFM. TopUML allows getting the most needed design models that conform to TFM (Osis et al., 2014). Thus, topological modeling together with TopUML application covers both analysis and design of the planned information system. Author of the survey feels that TopUML could become more useful if tool support was developed for it, i.e., formal transformations between models can and should be automated for higher efficiency of the approach. Also, no research has been done on the application of TopUML for platform independent viewpoint transformation into platform specific viewpoint and generating software code.

5 IDM APPROACH

The Integrated Domain Modeling (IDM) is a research subfield of TFM4SE. The aim of this research is to find a way of formalizing the knowledge about business system, and to get TFM from this knowledge. In addition, tool support is developed for IDM approach. In this subsection, articles which relate to IDM approach are reviewed. The chief researcher is A. Slihte, and his supervisor is J. Osis.

Paper (Slihte, 2009) concentrates on development of a software tool for TFM construction. For this purpose, MOF-compatible metamodel of TFM is introduced, and it is used by the tool. Tool prototype is developed. It allows manually creating a TFM. This is an important milestone in the development of TFM4SE. In the paper, for the first time TFM for MDA approach is called "TFM4MDA". This name is used nowadays.

In papers (Slihte, 2010) and (Osis and Slihte, 2010), an approach for automatic creation of TFM from the knowledge about a system is proposed (not yet developed). The knowledge is represented by business use cases – a formal data structure that can be processed automatically. A proposal to use natural language processing to analyze sentences of business use cases is made. This way, functional features are obtained. Although, not all attributes of functional feature tuple can be obtained automatically – only action (*A*), object (*O*), result (*R*) and preconditions (*PrCond*). By analyzing business use cases, it is possible to obtain topology of TFM. However, some cause-effect relations (elements of TFM's topology) need to be added manually. To sum up, the process of getting TFM from business use cases can be automated, but it requires minor interaction with system analyst.

In order to make business use cases "understandable" for computer, the step sentences are defined using a controlled natural language. In particular, *Attempto Controlled English* is used (Slihte et al., 2011). However, there are some problems with business use cases: 1) ambiguity, e.g., possibility to express the same meaning using different words; 2) inconsistency of business use cases, i.e., there might be steps defined that do not make sense in the given business system. To solve these problems, it is proposed to use ontology. Ontologies provide logical statements that describe what terms are and how they are related to each other. Ontology allows validation of procedural knowledge about a system which is represented by business use cases. Ontology and use cases need to be modified iteratively until they correspond. Then, it is possible to construct a TFM. Nevertheless, TFM also has to be validated. If any changes are necessary, they will have to be done in the ontology and business use cases, and then the TFM can be regenerated.

Paper (Slihte et al., 2011) reviews declarative and procedural knowledge. To represent declarative knowledge ontology is used, and to represent procedural knowledge – business use cases are applied. Paper also reviews technical opportunities of mapping the knowledge. Ontology can be developed using *OWL* standard. *Attempto Controlled English* for business use cases is mentioned in (Slihte et al., 2011). *Attempto Parsing Engine* can be used for natural language processing.

In paper (Osis et al., 2012), MOF-compatible metamodel of business use cases is introduced. A tool that uses this metamodel and allows constructing business use case model is developed.

Paper (Slihte and Osis, 2014) gives a name for an approach that is reviewed in this subsection: the Integrated Domain Modeling (IDM). In paper, a case of successful application of IDM approach in real business project is studied. A tool for automatic transformation "Business use cases → TFM" is developed. IDM approach is published in A. Slihte's PhD thesis (Slihte, 2015). This is a very important milestone in the development of TFM4SE.

To sum up, IDM approach concentrates on pre-CIM analysis, and also introduces formalism to this early stage of software development. The approach provides theory and toolset for obtaining TFM (which is a formal CIM in MDA framework) from the formalized knowledge about a business system.

The toolset allows creating business use case model, creating ontology, creating TFM (manually), and automatically obtaining TFM from business use cases. Nevertheless, author of this survey sees a gap in functionality of the mentioned toolset – correspondence between ontology and business use cases is not validated automatically.

6 FURTHER DEVELOPMENT OF TOPOLOGICAL MODELING

Paper (Osis and Silins, 2009) introduces an approach for development of embedded systems. It combines principles of co-design and MDA. Name of the approach is "Topological Function – Architecture Co-Design".

Research in paper (Asnina and Osis, 2010) concentrates on bridging problem and solution domains. As before, TFM is used as a formal bridging mechanism. Paper introduces new names for TFM types: "as is" TFM – TFM of problem domain; 2) "to be" TFM – TFM of solution domain; 3) TFM of information system. Information system is a subsystem of solution domain. Hence, TFM of information system can be separated from TFM of solution domain with closure operation. Compliance between a problem domain and a solution domain is proved if continuous mapping between "as-is" TFM and "to-be" TFM is in place. Likewise, continuous mapping must be kept between "to-be" TFM and TFM of information system. (Closure operation and continuous mapping are fundamentals of TFM (Osis, 1969).)

In paper (Osis and Asnina, 2011a), authors reason about the state of software development. They share with some other experts, e.g., C. Jones, the opinion that the way software is built is primitive. To become better, software *development* must turn into software *engineering*. The word "engineering" intends a theory approved, completely realized and reused many times in practice that gives a qualitative and relatively inexpensive end product in accurately predictable timeframes. In order to satisfy high effectiveness and quality of the software development we need *Theory Driven Architecture* and *Scientific Software Engineering*. Paper's title is "Is Modeling a Treatment for the Weakness of Software Engineering?" The answer is positive, but if and only if modeling is based on mathematical formalism from the very beginning of software development process.

An engineering model should satisfy five key characteristics: abstraction, understandability, accuracy, predictability and inexpensiveness, as it is stated in (Osis and Asnina, 2011b). TFM satisfies them. Thus, it is concluded that TFM is an engineering model. TFM is compared to Petri nets in the paper. Both models are formal. TFM has an advantage over Petri nets – TFM is applicable in modeling complex systems, while Petri nets as self-sufficient model is not quite convenient.

In paper (Asnina and Osis, 2011), it is concluded that CIM may include three main parts: 1) CIM – Knowledge Model (or pre-CIM); 2) CIM – Business Model; 3) CIM – Business Requirements for the System. Within TFM4MDA: CIM – Knowledge Model is informal verbal description of a system, or, if IDM is used, it is Ontology with Business use cases; CIM – Business Model is TFM; CIM – Business Requirements for the System are Use cases obtained from TFM. Also, informal guidelines of how to derive business processes from TFM are introduced in the paper.

Paper (Osis and Asnina, 2011c) describes benefits and limitations of use case techniques. New formal guidelines for obtaining use cases from TFM are introduced.

Paper (Asnina et al., 2011) demonstrates the establishment of formal trace links to real world functional units and entities from user requirements and analysis artifacts. These links show element interdependence make the impact analysis more thorough. Thanks to the formalism of TFM and to formal transformations from TFM to other models, it is possible to automate the mentioned tracing.

In TFM, the combinations of causes might exist that are sufficient or both necessary and sufficient to cause an appearance of the effect. To specify these combinations, logic is introduced to TFM in (Asnina et al., 2012). This approach of representing logic in TFM is different to the one described in (Donins, 2012a), and serves different purpose. A formal specification of TFM's cause-effect relation is introduced – a 4-tuple:

$$<C, E, N, S>,$$

where C is a cause functional feature, E is an effect functional feature, N is the necessity of the functional feature C for generating the functional feature E, and S is sufficiency of C for generating E. There was no formal specification for cause-effect relation before. In addition, logical operators are introduced: conjunction (AND), disjunction (OR, XOR) and negation (¬).

In paper (Asnina et al., 2013), an attribute is added to the tuple that specifies cause-effect relation:

$$<C, E, N, S, \underline{Refs}>,$$

where *Refs* (references) is a set of unique tuples <*Ref_Ids*, *LOp*>, where *LOp* is a logical operation (e.g., AND, OR), and *Ref_Ids* is a set of tuples <*C**, *E**> of cause-and-effect relations (<*C*, *E*> is not equal to <*C**, *E**>) that participate in logical operation *LOp* together. Also, paper introduces transformation "TFM → UML Activity diagram". This transformation, thanks to integrated logic, is more precise than the one defined in (Osis et al., 2007c). This transformation is an alternative to the one that is proposed in (Donins, 2012a), that uses logical relations instead of cause-effect relations specified by tuple <*C*, *E*, *N*, *S*, *Refs*>.

In paper (Asnina and Ovcinnikova, 2015), formal specifications of TFM elements are refined. *Refs* attribute is removed from the cause-and-effect specification. New tuple that describes cause-effect relation:

$$< ID, X_c, X_e, N, S>,$$

where *ID* is a unique identifier of a relation, X_c is a cause functional feature, and X_e is an effect functional feature. So it is quite the same as it was defined in (Asnina et al., 2012). The exclusion of *Refs* attribute leads to moving logical combination to the definition of functional feature. New tuple that describes functional feature:

$$< A, R, O, PrCond, PostCond, Pr, Ex, \underline{InRel}, \underline{OutRel}>,$$

where *InRel* determines combinations of possible logical relations among incoming cause-effect relations, and *OutRel* – among outgoing cause-effect relations. *Pr* and *Ex* are also new attributes, but are not important in the context of logical combinations. *Pr* is a set of responsible entities (systems or subsystems) which provide or suggest the action with the set of certain objects. *Ex* is a set of responsible entities which enact the action. Finally, the specification of logical relation is given (same as in (Donins, 2012a)):

$$< ID, T, R_T>,$$

where *ID* is a unique identifier of a relation, *T* is a set of cause-effect relations that participate in this logical relation, and R_T is a logical operator AND, OR, or XOR over T. So paper (Asnina and Ovcinnikova, 2015) contains all newest formal definitions of TFM elements. In addition, the research of the paper tries to find a mechanism for specifying business rules that would be appropriate for TFM. The conclusion is made that the best option is *Decision Model and Notation* – a standard proposed by the OMG for business process modelling. Nevertheless, the result of the research needs to be validated for cases where systems have the complex behavior.

Paper (Osis and Asnina, 2015) supplements paper (Osis and Asnina, 2011a). *Software Engineering Method and Theory* (SEMAT) group is described in more detail. This group supports a principle to base software engineering on a solid theory and best practices. The idea is to create a solid theory that is a basis for the Kernel language that formalizes different software development methods. The Kernel language is being created as an OMG standard.

7 CONCLUSION

In this paper, the development of topological modeling for software development was reviewed. Since 2004, TFM is used as a computation independent model in the framework of MDA. TFM4MDA approach – its theoretical basis, tool support and cogency of usefulness for software development – is a result of the reviewed research.

TFM4SE, including TopUML application and IDM approach, covers both analysis and design of the planned information system. IDM introduces formalism to the pre-CIM analysis. TopUML supports the obtaining of design models on PIM and PSM levels that conform to the formal CIM – TFM.

Author derives the subfields of topological modeling that were researched: Theoretical basis of TFM; Theory to ensure compliance between problem domain and solution domain; Mapping of requirements onto TFM; Development of mechatronic and embedded systems; Metamodeling; Model Driven Architecture; Construction of problem domain model from knowledge about the system; Obtaining UML diagrams from TFM; Modeling of business processes; Tool support for the approach.

For all mentioned subfields, author of the survey does not see any significant lacks or contradictions in the developed theory. The reviewed articles successfully expose the theory. However, TFM4SE lacks tool support. There are unimplemented possibilities of automation: 1) validation of correspondence between ontology and business use cases in IDM; 2) formal transformations between models in TopUML; 3) tracing from analysis artifacts to real world functional units and entities.

The supporting tool would make the approach more efficient. Thus, development of tool may be one of the directions of future work.

TFM4MDA does not cover the transformation of design models on PIM level to models on PSM level. This also might be a direction for future research.

Overall, author thinks that TFM4SE is ready to be used in industry as an approach of software development. Author believes that TFM is cogent.

REFERENCES

Alksnis, G., Asnina, E., Osis, J., Silins, J. 2005, Formalization of Software Development: Problems and Solutions. In: *Applied computer systems*. Volume 22, pp.204-216. ISSN 1407-7493.

Alksnis, G., Osis, J. 2002, Formalization of Software Engineering by Means of the Theory of Categories. In: *Scientific Proceedings of Riga Technical University*. Series – Computer Science (5), Volume 13, Riga, RTU, pp. 157-163.

Alksnis, G., Osis, Ya. 2001, Category Theory and Computer Science. In: *Computer Science, Applied Computer Systems, Series – Computer Science (5), Vol. 8, Scientific Proceedings of Riga Technical University, Riga*, pp. 59-67.

Asnina, E. 2006, The Formal Approach to Problem Domain Modelling Within Model Driven Architecture. In: *Proceedings of the 9th International Conference on Information Systems Implementation and Modelling (ISIM'06), Czech Republic, Přerov, 25-26 April, 2006*. Ostrava: Jan Štefan MARQ, pp.97-104. ISBN 80-86840-19-0.

Asnina, E., Gulbis, B., Osis, J., Alksnis, G., Donins, U., Slihte, A. 2011, Backward Requirements Traceability within the Topology-based Model Driven Software Development. In: *Proceedings of the 3rd International Workshop on Model-Driven Architecture and Modeling-Driven Software Development (MDA & MDSD 2011), China, Beijing, 7-11 June, 2011*. Lisbon: SciTePress, pp.36-45. ISBN 9789898425591.

Asnina, E., Osis, J. 2002, Formalization Problems and Perspectives of the Program Development (in Latvian). In: *Scientific Proceedings of Riga Technical University*. Series – Computer Science (5), Volume 13, Riga, RTU, pp. 145-156.

Asnina, E., Osis, J. 2006, The Computation Independent Viewpoint: a Formal Method of Topological Functioning Model Constructing. In: *Applied computer systems*. Vol.26, pp.21-32. ISSN 1407-7493.

Asnina, E., Osis, J. 2008, Analysis of Multifractal System Properties in Object-Oriented Software Development. In: *Applied computer systems*. Vol.34, pp.37-45. ISSN 1407-7493.

Asnina, E., Osis, J. 2010, Computation Independent Models: Bridging Problem and Solution Domains. In: *Proceedings of the 2nd InternationalWorkshop on Model-Driven Architecture and Modeling Theory-Driven Development (MDA & MTDD 2010), in conjunction with ENASE 2010, Greece, Athens, 22-24 July, 2010*. Lisbon: SciTePress, pp.23-32. ISBN 9789898425164.

Asnina, E., Osis, J. 2011, Topological Functioning Model as a CIM-Business Model. In: *Model-Driven Domain Analysis and Software Development: Architectures and Functions*. IGI Global, Hershey - New York, pp. 40 – 64. Available from: doi: 10.4018/978-1-61692-874-2.ch003

Asnina, E., Osis, J., Jansone, A. 2012, System Thinking for Formal Analysis of Domain Functioning in the Computation Independent Model. In*: Proceedings of the 7th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2012), Poland, Wrocław, 29-30 June, 2012*. Lisbon: SciTePress, pp.232-240. ISBN 9789898565136.

Asnina, E., Osis, J., Jansone, A. 2013, Formal Specifications of Topological Relations. In: *Databases and Information Systems VII: Selected Papers from the Tenth International Baltic Conference (DB&IS 2012), Lithuania, Vilnius, 8-11 July, 2012*. Amsterdam: IOS Press, pp.175-188. ISBN 978-1-61499-160-1. e-ISBN 978-1-61499-161-8. Available from: doi:10.3233/978-1-61499-161-8-175

Asnina, E., Osis, J., Kirikova, M. 2008, Design of Fractal-Based Systems Within MDA: Platform Independent Modelling. In: *Proceedings of the Third AIS SIGSAND European Symposium on Analysis, Design, Use and Societal Impact of Information Systems (SIGSAND-EUROPE 2008), Germany, Marburg, 12-13 June, 2008*. Marburg: Koellen-Verlag, pp.39-53. ISBN 978-3-88579-223-9. ISSN 1617-5468.

Asnina, E., Ovcinnikova, V. 2015, Specification of Decision-making and Control Flow Branching in Topological Functioning Models of Systems. In: *Proceedings of 10th International Conference on Evaluaton of Novel Approaches to Software Engineering, Spain, Barcelona, 29-30 April, 2015*. Portugal: SciTePress, pp.364-373. ISBN 978-989-758-100-7.

Birgelis, J., Osis, J. 2005, Generalization of MDA and Software Synthesis. In: *Scientific Proceedings of Riga Technical University*. Series – Computer Science (5), Volume 22, Riga, RTU, pp. 217-228.

Donins, U. 2010, Software Development with the Emphasis on Topology. In: *Advances in Databases and Information Systems: Lecture Notes in Computer Science*. Volume 5968, Berlin: Springer Berlin Heidelberg, pp. 220-228. ISBN 9783642120817.

Donins, U. 2012 a, Semantics of Logical Relations in Topological Functioning Model. In: *Proceedings of the 7th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2012), Poland, Wrocław, 29-30 June, 2012*. Lisbon: SciTePress, pp.217-223. ISBN 9789898565136.

Donins, U., Osis, J. 2009, Reconciling Software Requirements and Architectures within MDA. In:

Applied computer systems. Volume 38, pp. 84-95. ISSN 1407-7493.

Donins, U., Osis, J. 2011, Topological Modeling for Enterprise Data Synchronization System: A Case Study of Topological Model-Driven Software Development. In: *Proceedings of the 13th International Conference on Enterprise Information Systems. Vol.3, China, Beijing, 8-11 June, 2011*. Beijing: SciTePress, pp.87-96. ISBN 9789898425553.

Donins, U., Osis, J., Asnina, E., Jansone, A. 2012 a, Formal Analysis of Objects State Changes and Transitions. In: *Proceedings of the 7th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2012), Poland, Wrocław, 29-30 June, 2012*. Lisbon: SciTePress, pp.249-256. ISBN 9789898565136.

Donins, U., Osis, J., Asnina, E., Jansone, A. 2012 b, Using Functional Characteristics to Analyze State Changes of Objects. In: *Databases and Information Systems. Tenth International Baltic Conference on Databases and Information Systems: Local Proceedings, Materials of Doctoral Consortium, Lithuania, Vilnius, 8-11 July, 2012*. Vilnius: Žara, pp.94-106. ISBN 9789986342748.

Donins, U., Osis, J., Slihte, A., Asnina, E., Gulbis, B. 2011, Towards the Refinement of Topological Class Diagram as a Platform Independent Model. In: *Proceedings of the 3rd International Workshop on Model-Driven Architecture and Modeling-Driven Software Development (MDA & MDSD 2011), China, Beijing, 8-11 June, 2011*. Lisbon: SciTePress, pp.79-88. ISBN 9789898425591.

Donins, Uldis. 2012 b, *Topological Unified Modeling Language: Development and Application*. PhD Thesis. Riga: [RTU]. 224 p.

Gelfandbain, J., Osis, J., Markovich, Z., Novozilova, N. 1990, Diagnostics on Graph Models (in Russian). In: *Proceedings of conference on gas turbine engines, Moscow-Harkow*, pp. 34 – 35.

Ivasiuta, O., Osis, J. 1999, Methodics to Comparing Software Design Methodologies. In: *Proc.of the 33rd International Conference "Modelling and Simulation of Systems" (MOSIS'99). Roznov pod Radhostem, Czech Republi*, V. ISM'99, pp. 67-74.

Miller, J., Mukerji, J. 2003, *MDA Guide Version 1.0.1*, OMG, viewed 10 September 2015, <http://www.omg.org/cgi-bin/doc?omg/03-06-01>

Nahimova, M., Osis, J. 2002, Specific Features of the Modeling of Mechatronic Systems (in Latvian). In: *Scientific Proceedings of Riga Technical University*. Series – Computer Science (5), Volume 13, Riga, RTU, pp. 181-189.

Osis, J. 1969, Topological Model of System Functioning (in Russian). In: *Automatics and Computer Science, J. of Academia of Siences, Riga, Latvia, Nr. 6*, pp. 44-50.

Osis, J. 1972, *Diagnostics of Complex Systems* (in Russian). Summary of Habilitation Thesis, Latvian Academy of Sciences, Riga, 56 p.

Osis, J. 1991, Topological Models in Technical and Medical Diagnostics, in Image Recognition and in Expert Systems in Latvia (in Latvian). In: *World Congress of Latvian Scientists, Vol. 5. Riga*.

Osis, J. 1997, Development of Object-Oriented Methods for Hybrid System Analysis and Design. In: *Proceed. of the 23rd Conference of the ASU. Stara Lesna, Slovakia*, pp. 162-170

Osis, J. 2001 a, RTU Scientific School of System Modeling (in Latvian). In: *Computer Science, Applied Computer Systems, Series – Computer Science (5), Vol. 8, Scientific Proceedings of Riga Technical University, Riga*, pp. 6-17.

Osis, J. 2001 b, What is the Precise Relationship between Domain Modelling and Architectural Design and Modelling? In: *Proceedings of the 4th ECOOP Workshop on Object-Oriented Architectural Evolution, 15th ECOOP, Budapest, Hungary, 18–22 June*, pp. 9-12.

Osis, J. 2001 c, Brief Survey of Object-Oriented Approach. In: *Computer Science, Applied Computer Systems, Series – Computer Science (5), Vol. 8, Scientific Proceedings of Riga Technical University, Riga*, pp. 23-32.

Osis, J. 2001 d, Object-Oriented System Analysis – Case Processing or Construction of Formalism? (in Latvian). In: *Proceedings of Plenary session of the 2nd World Congress of Latvian Scientists, Riga*, pp. 596.

Osis, J. 2001 e, RTU Scientific School of Modeling and Studying of Complex Systems (in Latvian). In: *Proceedings of the 42nd International Conference of Riga Technical University, Riga, RTU*, pp. 38 – 40.

Osis, J. 2003 a, Extension of Software Development Process for Mechatronic and Embedded Systems. In: *Proceedings of the 32nd International Conference on Computers and Industrial Engineering, University of Limerick, Ireland, 11 –13 August*, pp. 305 – 310.

Osis, J. 2003 b, Topological Functioning Model Support for Software Engineering. In: *Scientific Proceedings of Riga Technical University*. Series – Computer Science (5), Volume 17, Riga, RTU, pp. 31 – 42.

Osis, J. 2004, Software Development with Topological Model in the Framework of MDA. In: *Proceedings of the 9th CAiSE/IFIP8.1/EUNO International Workshop on Evaluation of Modeling Methods in Systems Analysis and Design (EMMSAD'2004) in connection with the CAiSE'2004*. Volume 1, RTU, Riga, pp. 211-220.

Osis, J. 2006 a, Problem Domain Modeling at the Beginning of MDA Life Cycle By Means of Topological Functioning Model. In: *Proceedings of the 7th International Baltic Conference on Databases and Information Systems (DB&IS'06)*. Vilnius, Lithuania, pp. 105-116.

Osis, J. 2006 b, Formal Computation Independent Model within the MDA Life Cycle. In: *International Transactions on Systems Science and Applications*. ISSN 1751-1461 (Print), ISSN 1751-147X (CD-ROM), V. 1, Nr. 2, Xiaglow Institute Ltd, Glasgow, UK, pp. 159-166.

Osis, J. 2006 c, Topological Functioning Model within the MDA Life Cycle. In: *Scientific Proceedings of Riga Technical University*. Series – Computer Science (5), Volume 26, Riga, RTU, pp. 9-20.

Osis, J., Asnina, E. 2008 a, Enterprise Modeling for Information System Development within MDA. In: *Proceedings of the 41st Hawaii International Conference on Systems Science (HICSS-41 2008), United States of America, Waikoloa, 7-10 January, 2008*. Waikoloa: IEEE Computer Society, pp.490-490. ISSN 1530-1605. Available from: doi:10.1109/HICSS.2008.150

Osis, J., Asnina, E. 2008 b, A Business Model to Make Software Development Less Intuitive. In: *International Conference on Innovation in Sofware Engineering (ISE 2008), Austria, Vienna, 10-12 December, 2008*. Vienna: IEEE Computer Society, pp.1240-1245. ISBN 9780769535142. Available from: doi:10.1109/CIMCA.2008.52

Osis, J., Asnina, E. 2011 a, Is Modeling a Treatment for the Weakness of Software Engineering? In: *Model-Driven Domain Analysis and Software Development: Architectures and Functions*. IGI Global, Hershey - New York, pp. 1-14. Available from: doi: 10.4018/978-1-61692-874-2.ch001

Osis, J., Asnina, E. 2011 b, Topological Modeling for Model-Driven Domain Analysis and Software Development: Functions and Architectures. In: *Model-Driven Domain Analysis and Software Development: Architectures and Functions*. IGI Global, Hershey - New York, pp. 15 – 39. Available from: doi: 10.4018/978-1-61692-874-2

Osis, J., Asnina, E. 2011 c, Derivation of Use Cases from the Topological Computation Independent Business Model. In: *Model-Driven Domain Analysis and Software Development: Architectures and Functions*. IGI Global, Hershey - New York, pp. 65 – 89. Available from: doi: 10.4018/978-1-61692-874-2.ch004

Osis, J., Asnina, E. 2015, Is Modeling a Treatment for the Weakness of Software Engineering? In: V.Díaz, J.Cueva Lovelle, B.García-Bustelo ed. *Handbook of Research on Innovations in Systems and Software Engineering*. Hershey, PA: IGI Global, pp.411-427. ISBN 9781466663596. e-ISBN 9781466663602. Available from: doi:10.4018/978-1-4666-6359-6

Osis, J., Asnina, E., Donins, U., Garcia-Diaz, V. 2014, Dependencies among Architectural Views Got from Software Requirements Based on a Formal Model. In: *Applied Computer Systems*. Vol.16, pp.5-12. ISSN 2255-8683. e-ISSN 2255-8691. Available from: doi:10.1515/acss-2014-0007

Osis, J., Asnina, E., Grave, A. 2007 a, Formal Computation Independent Model of the Problem Domain within the MDA. In: *Proceedings of the 10th International Conference on Information System Implementation and Modeling (ISIM 2007), Czech Republic, Hradec nad Moravici, 23-25 April, 2007*. Hradec and Moravici: Jan Štefan MARQ, pp.47-54.

Osis, J., Asnina, E., Grave, A. 2007 b, MDA Oriented Computation Independent Modeling of the Problem Domain. In: *Proceedings of the 2nd International Working Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2007), Spain, Barcelona, 23-25 July, 2007*. Barcelona: INSTICC Press, pp.66-71. ISBN 978-989-8111-10-4.

Osis, J., Asnina, E., Grave, A. 2007 c, Computation Independent Modeling within the MDA. In: *Proceedings of IEEE International Conference on Software, Science, Technology & Engineering (SwSTE07), Israel, Herzlia, 30-31 October, 2007*. Herzlia: IEEE Computer Society, Conference Publishing Services (CPS), pp.22-34. ISBN 978-0-7695-3021-5. Available from: doi:10.1109/SwSTE.2007.20

Osis, J., Asnina, E., Grave, A. 2008 a, Computation Independent Representation of the Problem Domain in MDA. In: *e-Informatica Software Engineering Journal*, Vol.2, Iss.1, pp. 29.-46, ISSN 1897-7979.

Osis, J., Asnina, E., Grave, A. 2008 b, Formal Problem Domain Modeling within MDA. In: *Communications in Computer and Information Science (CCIS). Software and Data Technologies: Second International Conference ICSOFT/ENASE 2007: Revised Selected Papers, Germany, Berlin, 22-25 July, 2007*. Berlin: Springer-Verlag Berlin Heidelberg, pp.387-398. ISBN 9783540886549. e-ISBN 9783540886556. ISSN 1865-0929. Available from: doi:10.1007/978-3-540-88655-6_29

Osis, J., Beghi, L. 1997, Topological Modelling of Biological Systems. In: *Proccedings of the third IFAC Symposium on Modelling and Control in Biomedical Systems (Including Biological Systems)*, D. A. Linkens, E. R. Carson (editors), Pergamon-Elsevier Science Publishing, Oxford, UK, pp. 337-342.

Osis, J., Donins, U. 2009 a, An Innovative Model Driven Formalization of the Class Diagrams. In: *Proceedings of 4th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2009), Italy, Milan, 9-10 May, 2009*. Milan: INSTICC Press, pp. 134-145.

Osis, J., Donins, U. 2009 b, Modeling Formalization of MDA Software Development at the Very Beginning of Life Cycle. In: *Advances in Databases and Information Systems: 13th East-European Conference (ADBIS 2009) : Associated Workshops and Doctoral Consortium : Local Proceedings, Latvia, Rīga, 7-10 September, 2009*. Riga: RTU, pp.48-61. ISBN 9789984301631.

Osis, J., Donins, U. 2010 a, Formalization of the UML Class Diagrams. In: *Evaluation of Novel Approaches to Software Engineering: 3rd and 4th International Conferences ENASE 2008/2009: Revised Selected Papers, Italy, Milan, 9-10 May, 2009*. Berlin: Springer-Verlag, pp.180-192. ISBN 9783642148187. e-ISBN 9783642148194. ISSN 1865-0929. Available from: doi:10.1007/978-3-642-14819-4_13

Osis, J., Donins, U. 2010 b, Platform Independent Model Development by Means of Topological Class

Diagrams. In: *Model-Driven Architecture and Modeling Theory-Driven Development: Proceedings of the 2nd International Workshop on Model-Driven Architecture and Modeling Theory-Driven Development (MDA & MTDD 2010), Greece, Athens, 22-24 July, 2010*. Lisbon: SciTePress, pp.13-22. ISBN 9789898425164.

Osis, J., Gelfandbain, J., Markovich, Z., Novozilova, N. 1991, *Diagnostics on Graph Models (on the Examples of Aviation and Automobile Technology)* (in Russian). Moscow, Transport, 244 p.

Osis, J., Merkuryev, Y., Ginters, E., Teilans, A. 1996, Object Oriented Modelling and Simulation Using LATISS. In: *Proceedings of the 22nd Conference of the ASU, Clermont-Ferrand, France, July*, pp.136-145.

Osis, J., Silins, J. 2002, Specifics of Modelling of Embedded Systems. In: *Scientific Proceedings of Riga Technical University.* Series – Computer Science (5), Volume 13, Riga, RTU, pp. 173-180.

Osis, J., Silins, J. 2009, Topological Function-Architecture Co-Design of Embedded Systems. In: *Advances in Databases and Information Systems: 13th East-European Conference (ADBIS 2009): Associated Workshops and Doctoral Consortium : Local Proceedings, Latvia, Rīga, 7-10 September, 2009*. Riga: RTU, pp.424-431. ISBN 9789984301631.

Osis, J., Slihte, A. 2010, Transforming Textual Use Cases to a Computation Independent Model. In: *Model-Driven Architecture and Modeling Theory-Driven Development: Proceedings of the 2nd International Workshop (MDA & MTDD 2010), Greece, Athens, 22-24 July, 2010*. Lisbon: SciTePress, pp.33-42. ISBN 9789898425164.

Osis, J., Slihte, A., Jansone, A. 2012, Using Use Cases for Domain Modeling. In: *Proceedings of the 7th International Conference on Evaluation of Novel Approaches to Software Engineering (ENASE 2012), Poland, Wrocław, 29-30 June, 2012*. Lisbon: SciTePress, pp.224-231. ISBN 9789898565136.

Osis, J., Sukovskis, U., Teilans, A. 1997, Business Process Modeling and Simulation Based on Topological Approach. In: *Proceedings of the 9th European Simulation Symposium and Exhibition, Passau, Germany*, pp. 496-501.

Slihte, A. 2009, The Concept of a Topological Functioning Model Construction Tool. In: *Advances in Databases and Information Systems: 13th East-European Conference (ADBIS 2009) : Associated Workshops and Doctoral Consortium : Local Proceedings, Latvia, Rīga, 7-10 September, 2009*. Rīga: RTU, pp.476-484. ISBN 9789984301631.

Slihte, A. 2010, The Specific Text Analysis Tasks at the Beginning of MDA Life Cycle. In: *Databases and Information Systems Doctoral Consortium, Latvia, Riga, 5-7 July, 2010*. Riga: SIA "Latgales druka", pp.11-22. ISBN 9789984451893.

Slihte, A., Osis, J. 2014, The Integrated Domain Modeling: A Case Study. In: *Databases and Information Systems: Proceedings of the 11th International Baltic Conference (DB&IS 2014), Estonia, Tallinn, 8-11 June, 2014*. Tallinn: Tallinn University of Technology Press, pp.465-470. ISBN 978-9949-23-632-9. e-ISBN 978-9949-23-633-6.

Slihte, A., Osis, J., Donins, U. 2011, Knowledge Integration for Domain Modeling. In: *Proceedings of the 3rd International Workshop on Model-Driven Architecture and Modeling-Driven Software Development (MDA & MDSD 2011), China, Beijing, 8-11 June, 2011*. Lisbon: SciTePress, pp.46-56. ISBN 9789898425591.

Slihte, A., Osis, J., Donins, U., Asnina, E., Gulbis, B. 2011, Advancements of the Topological Functioning Model for Model Driven Architecture Approach. In: *Proceedings of the 3rd International Workshop on Model-Driven Architecture and Modeling-Driven Software Development (MDA & MDSD 2011), China, Beijing, 7-11 June, 2011*. Lisbon: SciTePress, pp.91-100. ISBN 9789898425591.

Slihte, Armands. 2015, *The Integrated Domain Modeling: an Approach & Toolset for Acquiring a Topological Functioning Model*. PhD Thesis. Riga: [RTU]. 224 p.