# FORMAL METHOD OF FUNCTIONAL MODEL BUILDING BASED ON GRAPH TRANSFORMATIONS

**Janis Grundspenkis**[(a)], **Ieva Zeltmate** [(b)]

[(a)(b)] Riga Technical University, Faculty of Computer Science and Information Technology, Department of Systems Theory and Design

[(a)] janis.grundspenkis@cs.rtu.lv, [(b)] ieva.zeltmate@cs.rtu.lv

**ABSTRACT**
Structural modelling approach has been developed to support systematic, domain model–based knowledge acquisition for model–based diagnosis problem solving. Structural models capture declarative, deep knowledge about system morphology and operation in normal conditions as well under faults. Models of morphological and functional structure are built mainly using knowledge of problem domain experts. Formal methods and algorithms for building model of a morphological structure and its transformation into a model of a functional structure have been developed to reduce the workload of human experts. The paper focuses on model transformation issues and represents new results allowing to extend previously developed method of formal transformations. Analysis of different cases of logic of input and output flows of system components is carried out and corresponding transformations defined. The developed formal method is demonstrated for functional model building of cooling system of internal combustion engine.

Keywords: Structural modelling, reasoning, model transformations, decision making

## 1. INTRODUCTION

Complex technical systems often have different types of components: software and hardware including electronic, mechanical, hydraulic, pneumatic, electric, and /or even chemical (Kurki 1995). A plethora of different approaches has been developed for modelling of complex technical systems with the purpose of their diagnosis. At the same time many methods and models are not suitable for complex technical systems with physically heterogeneous elements where various physical processes (exchange of information, energy, material flows, etc.) proceed because it is impossible to build mathematical (analytical) models for a system as a whole. Alternative approaches try to overcome this drawback proposing, for example, to build general qualitative models (Fang 1994, Grundspenkis 1997, Harandi and Lange 1990) or to integrate multiple model types (Abu-Hanna and Jansweier 1994, Grundspenkis and Jekabsons 2001). Among these approaches only few support some kind of reasoning. The goal of structural modelling (Grundspenkis 1997, Grundspenkis 1999) is to develop a framework for knowledge acquisition, representation and processing based on a set of related structural models. The advantages of structural models are their ability to capture both shallow (expert's experience) and deep knowledge (cause–consequence relationships between components), as well as to support inference of a new knowledge for decision making about systems operation (Grundspenkis 2004).

As a rule, knowledge acquisition is a bottleneck when knowledge–based systems are developed. Working with problem domain experts is time consuming and interpretation of obtained results isn't unambiguous. Structural modelling supports systematic domain model based knowledge acquisition. That allows to understand the structure and operations of system under consideration (Grundspenkis 1999). Moreover, already developed formal methods and algorithms allow to build topology of both models, namely a model of a morphological structure (MSM) and a model of a functional structure (FSM). Thus experts only need to define primitives of MSM (details are given in the next section) and a model is generated automatically. Transformation of a MSM into a FSM allows keeping consistency between models which often isn't possible when each model is built from scratch. The main idea of model transformations is based on formal method proposed in graph theory for undirected graphs. The essence of method is as follows: arcs of an initial graph (also called a node graph) are replaced by nodes of so called line graph (Tutte 2001). In structural modelling this method has been modified for directed graphs.

The paper presents a new transformation method in which logic of input and output flows of components of MSM are taken into account. The paper is organized as follows. Due to the fact that the paper pretty much is based on previous works brief descriptions of basic notions of MSM and FSM are given in sections 2 and 3, respectively. The section four is devoted to the analysis of different cases of logic of input and output flows. In result corresponding transformations are defined. An example which demonstrates the proposed transformation method is given, too. Conclusions and a short outline of future work are given in the last section.

## 2. THE BASIC NOTIONS OF MSM

Structural modelling approach and its applications are described in details in several papers (Grundspenkis 1997, Grundspenkis 1999, Grundspenkis 2002, Grundspenkis 2004). Due to the focus of the paper in this and in the next section only basic notations of models of morphological and functional structures are given. The first step of model building within the framework of structural modelling is identification of all available knowledge sources (human experts, handbooks, reports, graphics, etc.). The acquired knowledge is captured into small, independent, composable and decomposable units of knowledge which are visualized as objects (see Figure 1). Thus, objects are basic units of abstract model of MSM. These primitives have input and output contacts. If interpreted in an application domain, abstract objects correspond to the components of the given system, and contacts represent their inputs and outputs.

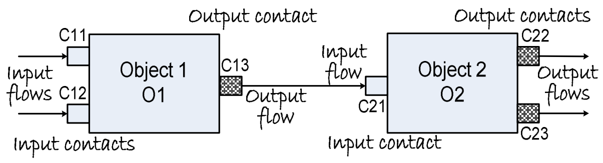

Figure 1: Representation of basic units of MSM

The connection of one object's output to another object's input is the only way how components can interact. Interactions are called flows. Human experts must list all components of a system, define all contacts and related flows. A topology of MSM is determined automatically by the *Automated Structural MOdelling System* (ASMOS) which maintains consistency between different levels of granularity of MSM by applying continuous mapping of models (Grundspenkis 1997).

The MSM supports so called structural reasoning that is based on the analysis of direct and indirect connectivity of objects. A direct connectivity is described by structural equation of the following form:

*flow$_1$ (IN) LOGOP flow$_2$ (IN) LOGOP … LOGOP flow$_N$ (IN) = flow$_1$ (OUT) LOGOP flow$_2$ (OUT) LOGOP … LOGOP flow$_M$ (OUT).*

Here LOGOP stands for logical operations AND or OR which define logic of flows. At the input side AND means that all connected flows are needed to get output, while OR means that only one flow is needed. At the output side AND means that all flows are produced by a component, but OR means that each output flow is produced in one operation mode. Thus, in case if objects are represented as graph nodes they have AND/AND, OR/AND, OR/OR, and AND/OR logic as it is shown in Figure 2. More complicated cases are discussed later.

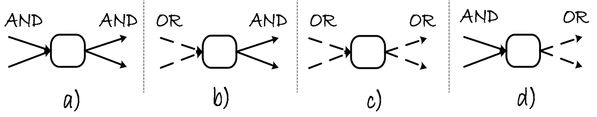

Figure 2: Logic of graph nodes

Composition of structural equations supports exploration of indirect connectivity and made possible conclusions about conditions of flow existence between inputs and outputs of different objects. In case of indirect connectivity the purpose of structural reasoning is twofold. First, it provides knowledge for decision making about requirements for the maintenance of designed operation modes. Second, results of structural reasoning are used to evaluate the degree of structural importance and the rank dispersion that support decision making about functionally overloaded components (Grundspenkis 1999).

## 3. THE BASIC NOTIONS OF FSM

To support reasoning about processes proceeding in physical components a model of functional structure (FSM) is used which encodes knowledge about functions of normally operating system, i.e., when a system under analysis has no faults. If visualized, the basic units of FSM are displayed as a diagram (see Figure 3).

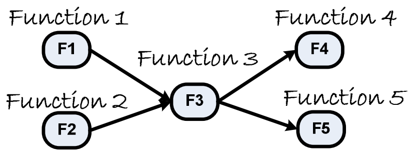

Figure 3: Representation of basic units of FSM

Expert's understanding of how a system works is organized as a representation that describes purpose why particular function is accomplished. More precisely, a function captures the intended purpose of an object (component), that is, a function specifies what a response is to a given stimulus, or, in other words, what is the result of proper behavior (functioning) of a given component. For example, the function of a water pump is "to provide a water flow under pressure". It is the purpose why a water pump is included in the system. This function in common with all others is represented as a node of FSM. Arcs of FSM represent cause–consequence relations between functions. Two notions – "”behavior" and "behavior state" of a contact of a given component are used to represent functions. Behavior is a characteristic of an input/output relation that explains how structure insures the fulfillment of functional specifications. In other words, behavior specifies an action that a component performs upon its "substance", i.e., it specifies how a reaction to a given stimulus is achieved. To provide a water flow under pressure, a water pump's behavior must be "pumping". Behavior state, in its turn, specify how flows "act" in corresponding inlets and outlets of components. Thus, each object described in MSM has its own behavior and object's contacts are mapped on corresponding behavior states. To conclude the description of basic notions used in a FSM, lets state that the behavior states at input and output contacts of a water pump are described as "water flows trough the inlet of water pump", and "water flows through the outlet of water pump", respectively. Analysis of FSM supports decision making about processes going on in physical components of the system.

## 4. TRANSFORMATION OF MSM TO A FSM

In the structural modelling framework the basic ideas of transformation of MSM to a FSM have been described in (Grundspenkis 1997). A topology of FSM is derived from a MSM using transformation of a node graph which represents a MSM into a corresponding line graph which represents a FSM. Actually this transformation differs from transformation described in graph theory (Tutte 2001). In structural modelling the following constraint on relationships between nodes of a line graph is defined: there is a relationship between nodes of a line graph if and only if the corresponding arcs of a node graph have the same direction. The difference between transformations defined in graph theory and in structural modelling is shown in Figure 4. A line graph in case *b* corresponds to the graph theory definition, but a line graph in case *c* corresponds to the FSM definition.

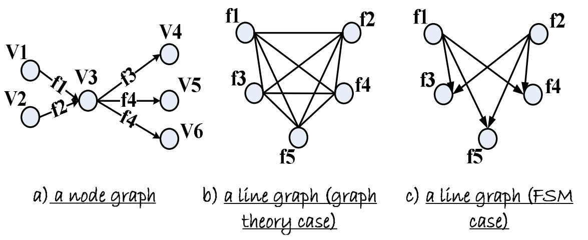
V1
V2
V3
V4
V5
V6
f1
f2
f3
f4
f5
a) a node graph
b) a line graph (graph theory case)
c) a line graph (FSM case)

Figure 4: An example of transformation

The defined constraint guarantees correct cause–consequence relationships between nodes of FSM which represent functions. It is worth to stress that automation of the model transformation significantly decreases the workload of experts because they are asked only to define semantics of nodes of FSM, not to build a FSM from scratch.

The rest of the paper is devoted to detailed analysis of all cases of transformations taking into consideration logic of flows of MSM. Specific graphical representations are used to represent the logical relations (see Figure 5).

In our approach square on the flow and symbol ',' atop of the flow are used to display the logical component AND. For example, see the first case in Figure 5 – the square on the flow and symbol ',' (between the members of the expression) atop of the flow specifies that F2 and F3 are connected with the logical component AND. Triangle and symbol ';' are used to display the logical component OR. For example, see the fourth case in the Figure 5. Small line (see seventh case) and symbols '(' and ')' are used to display the brackets in the logical expression.

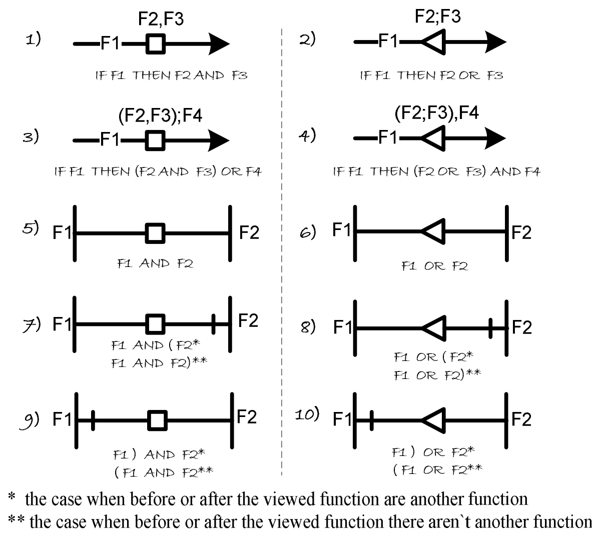
1) F2,F3 F1 IF F1 THEN F2 AND F3
2) F2;F3 F1 IF F1 THEN F2 OR F3
3) (F2,F3);F4 F1 IF F1 THEN (F2 AND F3) OR F4
4) (F2;F3),F4 F1 IF F1 THEN (F2 OR F3) AND F4
5) F1 F2 F1 AND F2
6) F1 F2 F1 OR F2
7) F1 F2 F1 AND ( F2* F1 AND F2)**
8) F1 F2 F1 OR ( F2* F1 OR F2)**
9) F1 F2 F1 ) AND F2* ( F1 AND F2**
10) F1 F2 F1 ) OR F2* ( F1 OR F2**
* the case when before or after the viewed function are another function
** the case when before or after the viewed function there aren`t another function

Figure 5: Graphical representations of logical relations

First four cases shown in Figure 5 demonstrate situations when one flow at the input side of an object affects many flows at the output. All other cases (from 5 to 10) specify situations when many flows at the input affect one or many flows at the output. The logical expression atop of the flow can be used in all cases.

Theoretically four basic combinations of input and output flows exist:

1. One flow at the object's input side and one at the output side.
2. One flow at the object's input side and many at the output side.
3. Many flows at the object's input side and one at the output side.
4. Many flows at the object's input side and many at the output side.

The first combination is simple and no specific notations are used to understand the logical cause–consequence relationship in a MSM and after transformation (see Figure 6). The logical expression for the case (a) is as follows: *IF F1 THEN F2*. One flow F1 is related to exactly one flow F2.

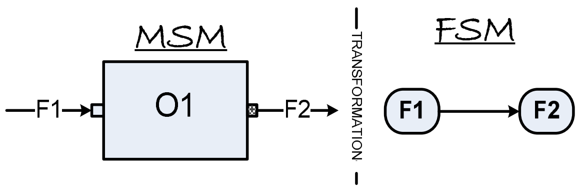

Figure 6: MSM and the corresponding FSM, case (a)

The second combination can be divided in 3 different cases. The first case is when one flow is at the input side and many flows linked with the logical operation AND are at the output side (see Figure 7).

The logical expression for the case (b) is as follows: *IF F1 THEN F2 AND F3*. One flow F1 creates more that one flow simultaneously - F2 and F3.

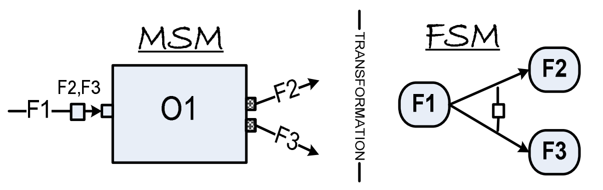

Figure 7: MSM and the corresponding FSM, case (b)

The second case differs from the first one only with the logical operation. One flow is at the input side and many flows linked with the logical operation OR are at the output side (see Figure 8).

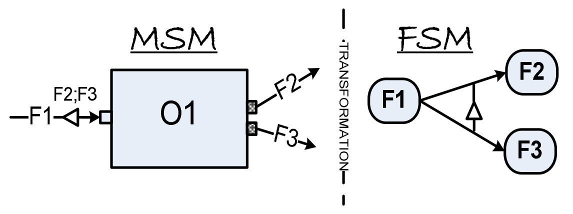

Figure 8: MSM and the corresponding FSM, case (c)

The logical expression for the case (c) is the following: *IF F1 THEN F2 OR F3*. One flow F1 can create one of two possible flows F2 or F3.

The third case is when one flow is at the input side and many flows linked with the logical operation AND/OR combinations are at the output side (see Figure 9). The logical expression for the case (d) is as follows: *IF F1 THEN (F2 OR F3) AND F4*. One flow F1 can create two flows simultaneously: flows F2 and F4 or in other mode flows F3 and F4.

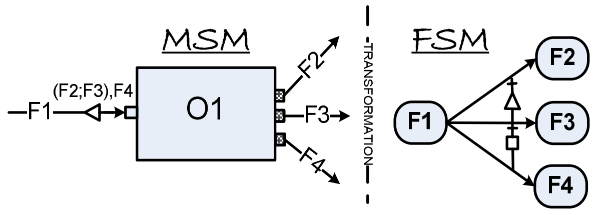

Figure 9: MSM and the corresponding FSM, case (d)

The third combination when many flows at the object input affect one flow at the output also can be divided in three different cases:

1. Many flows linked with the logical operation AND at the input side and one flow at the output side (see Figure 10).
2. Many flows linked with the logical operation OR at the input side and one flow at the output side (see Figure 11).
3. Many flows linked with the logical operation AND/OR combinations at the input side and one flow at the output side (see Figure 12).

The logical expression for the case (e) is the following: *IF F1 AND F2 AND F3 THEN F4*. Three flows F1 and F2 and F3 simultaneously can create exactly one flow F4.

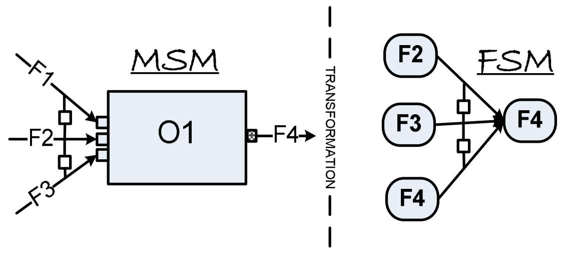

Figure 10: MSM and the corresponding FSM, case (e)

The logical expression for the case (f) is as follows: *IF F1 OR F2 OR F3 THEN F4*. One of the flows F1 or F2 or F3 in a fixed time can create one flow F4.

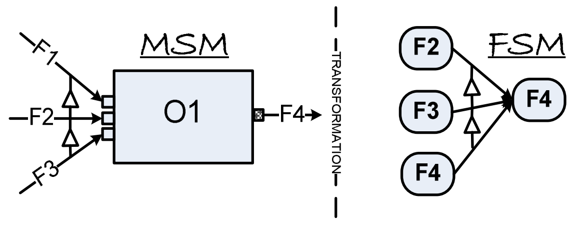

Figure 11: MSM and the corresponding FSM, case (f)

The logical expression for the case (g) is as follows: *IF F1 AND F2 AND F3 THEN F4*. Two flow combinations F1 and F3 or F2 and F3 can create one flow F4.

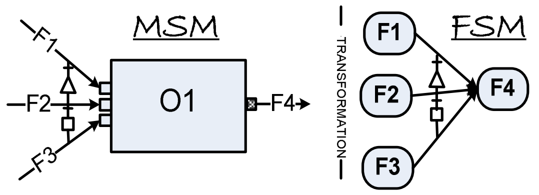

Figure 12: MSM and the corresponding FSM, case (g)

The fourth and the last combination of flows is the most complex and can be divided in many different cases, but we consider only six typical ones. The situation when many flows are linked with some logical operation at the output and all output flows are input of one object is not considered. In this combination the logical expression that can be obtained from a MSM after transformation to a FSM is decomposed in smaller rules. The first case is when many flows are linked with the logical operation AND at the input side and many flows are linked with the logical operation AND at the output side (see Figure 13).

The logical expression for the case (h) in the MSM and the FSM is: *IF F1 AND F2 AND F3 THEN F4 AND F5 AND F6*. Three flows F1 and F2 and F3 simultaneously create three flows F4 and F5 and F6.

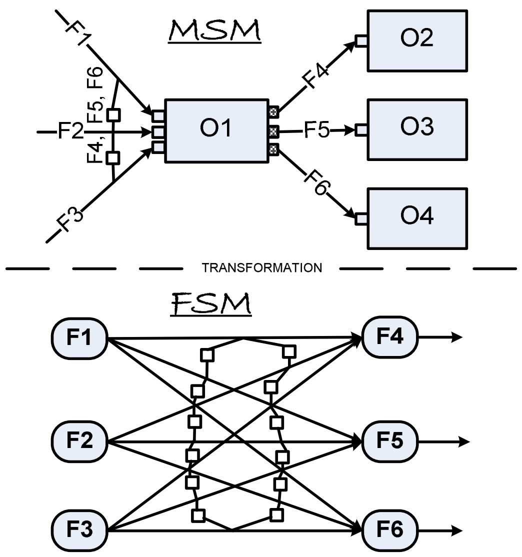

Figure 13: MSM and the corresponding FSM, case (h)

For the second case many flows are linked with the logical operation OR at the input side and many flows are linked with the logical operation OR at the output side (see Figure 14).

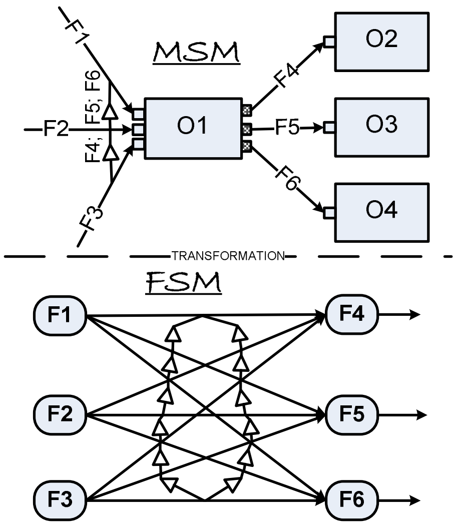

Figure 14: MSM and the corresponding FSM, case (i)

The logical expression for the case (i) in the MSM is: *IF F1 OR F2 OR F3 THEN F4 OR F5 OR F6.* One of the three flows F1 or F2 or F3 in a fixed time can create one of the three flows F4 or F5 or F6. The logical expressions in the FSM are:

- IF F1 THEN F4 OR F5 OR F6
- IF F2 THEN F4 OR F5 OR F6
- IF F3 THEN F4 OR F5 OR F6

or equally

- IF F1 OR F2 OR F3 THEN F4
- IF F1 OR F2 OR F3 THEN F5
- IF F1 OR F2 OR F3 THEN F6

The third case is when many flows are linked with the logical operation AND at the input side and many flows are linked with the logical operation OR at the output side (see Figure 15).

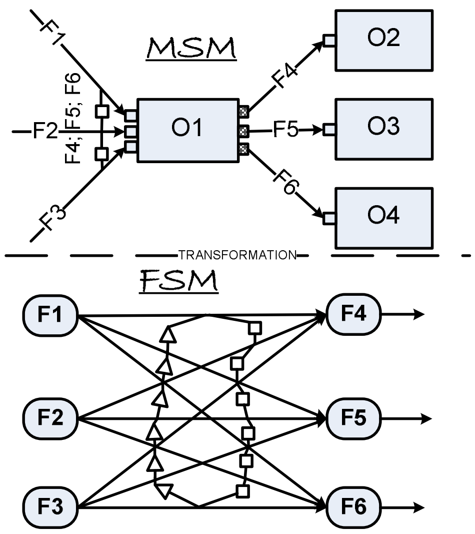

Figure 15: MSM and the corresponding FSM, case (j)

The logical expression for the case (j) in MSM is: *IF F1 AND F2 AND F3 THEN F4 OR F5 OR F6.* Three flows F1 and F2 and F3 simultaneously can create one of the three flows F4 or F5 or F6. The logical expressions in the FSM are:

- IF F1 AND F2 AND F3 THEN F4
- IF F1 AND F2 AND F3 THEN F5
- IF F1 AND F2 AND F3 THEN F6

For the fourth case many flows are linked with the logical operation OR at the input side and many flows are linked with the logical operation AND at the output side (see Figure 16).

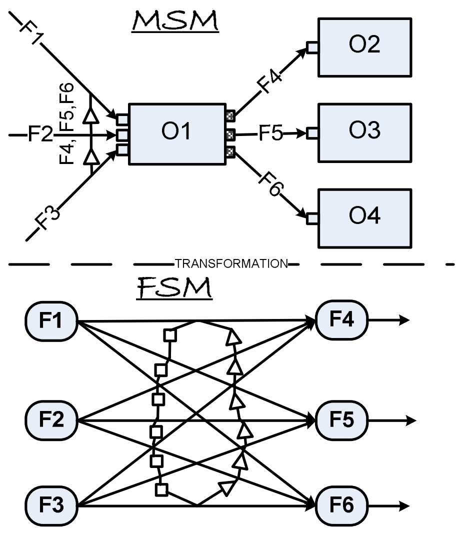

Figure 16: MSM and the corresponding FSM, case (k)

The logical expression for the case (k) in the MSM is: *IF F1 OR F2 OR F3 THEN F4 AND F5 AND F6.* One of the three flows F1 or F2 or F3 in a fixed time can create all three flows F4 and F5 and F6. The logical expressions in the FSM are:

- IF F1 THEN F4 AND F5 AND F6
- IF F2 THEN F4 AND F5 AND F6
- IF F3 THEN F4 AND F5 AND F6

The fifth case is when many flows are linked with the logical operation AND/OR combinations at the input side and many flows are linked with the logical operation AND/OR combinations at the output side (see Figure 17).

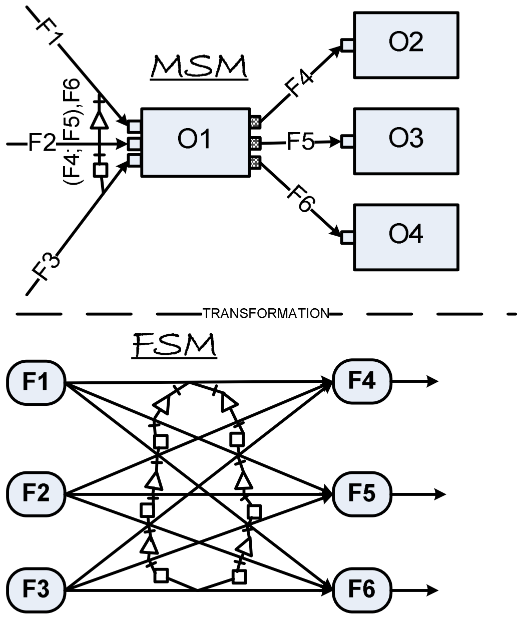

Figure 17: MSM and the corresponding FSM, case (l)

The logical expression for the case (l) in the MSM is: *IF (F1 OR F2) AND F3 THEN (F4 OR F5) AND F6.* Two flow combinations F1 and F3 or F2 and F3 can create one of the two flow combinations F4 AND F6 or F5 AND F6. The logical expressions in the FSM are:

- IF F1 AND F3 THEN (F4 OR F5) AND F6
- IF F2 AND F3 THEN (F4 OR F5) AND F6
- IF (F1 OR F2) AND F3 THEN F4 AND F6
- IF (F1 OR F2) AND F3 THEN F5 AND F6

The sixth case is the combination of the considered cases (from a to l) and for each of them in the MSM one rule can be acquired. As an example (see Figure 18) is displayed combination of cases (a) and (j). Many flows (F1, F2, and F3) are linked with the logical operation AND at the input side and affect many flows that are linked with the logical operation OR (F4, F5, F6) at the output. Besides, only the flow F1 affects the flow F7.

The logical expressions for the case (m) in the MSM are two. The first is: *IF F1 AND F2 AND F3 THEN F4 OR F5 OR F6.* Three flows F1 and F2 and F3 simultaneously can create one of the three flows F4 or F5 or F6. The second rule is: *IF F1 THEN F7.* The flow F1 creates also the flow F7. Corresponding logical expressions in the FSM are: IF F1 THEN F7 and all three described expressions that are used for the case (j) within model of functional structure.

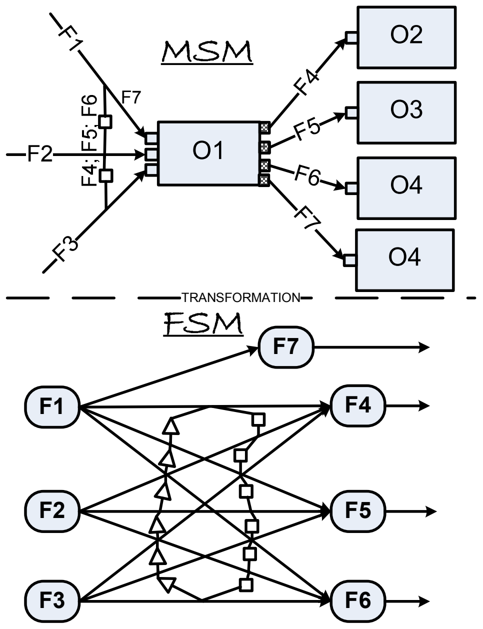

Figure 18: MSM and the corresponding FSM, case (m)

For better understanding of the defined transformations an example of a cooling system of an internal combustion engine (Grundspenkis 1999) is given. First, the model of a morphological structure is represented (see Figure 19).

For model of a morphological structure seven rules that correspond to the listed flow combination cases are defined:

1. IF F1 AND F13 THEN F2 (case e)
2. IF F2 THEN F3 OR F4 (case c)
3. IF F3 THEN F5 (case a)
4. IF F5 AND F8 THEN F6 AND F14 (case h)
5. IF F6 THEN F7 (case a)
6. IF F10 AND F11 AND F12 THEN F8 (case e)
7. IF F9 AND (F7 OR F4) THEN F1 (case g)

After transformations the model of functional structure of a cooling system (see Figure 20) is obtained.

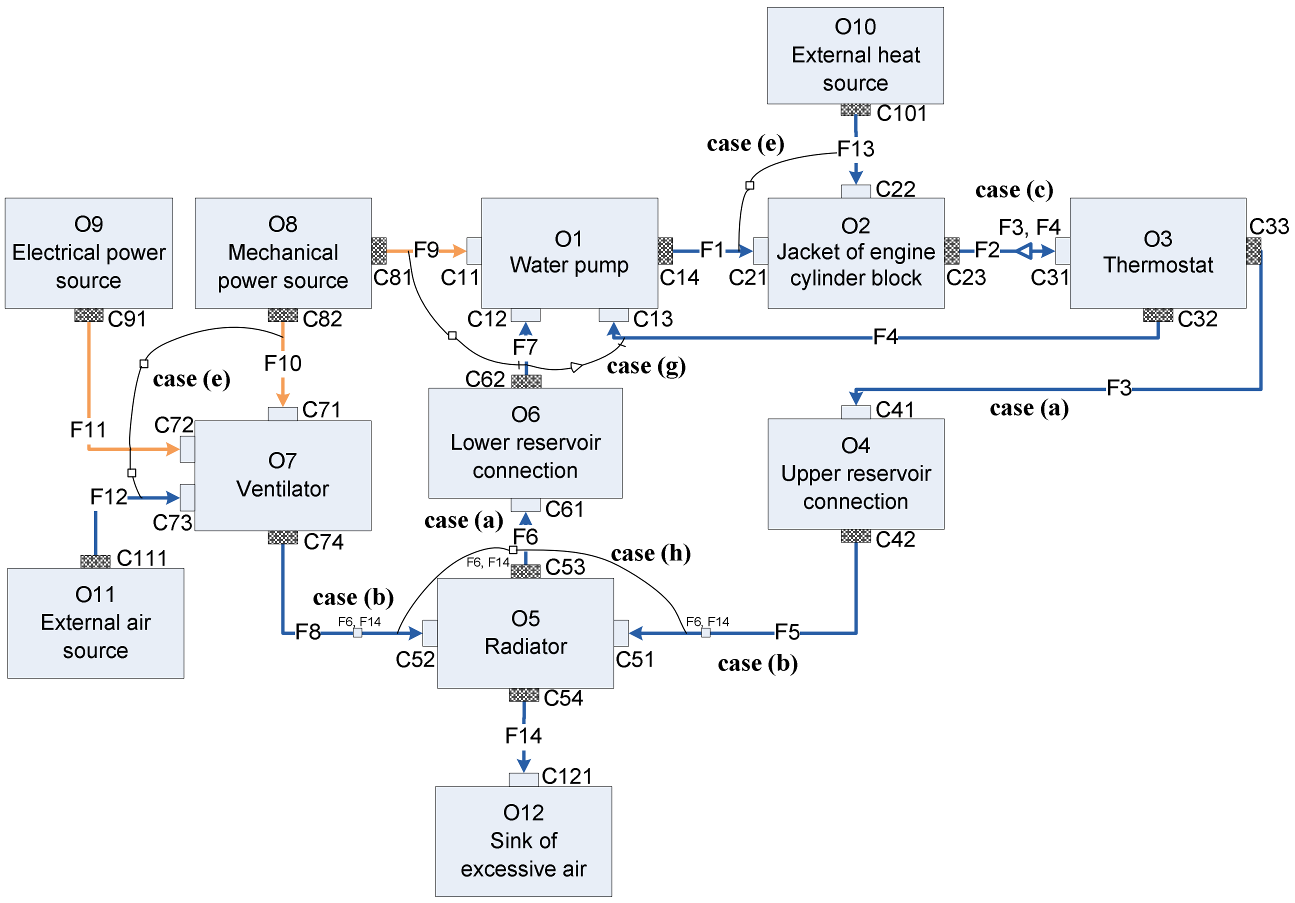

Figure 19: The morphological structure of a cooling system

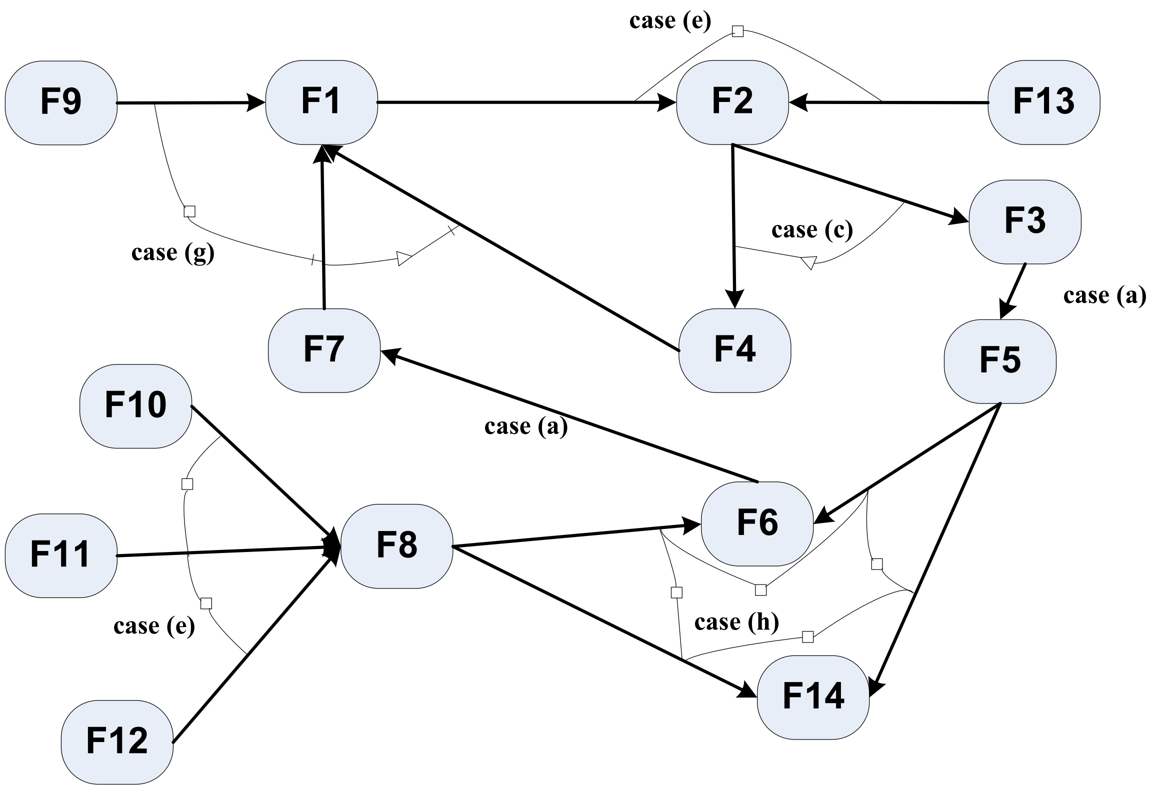

Figure 20: The functional structure of a cooling system

In the represented model of the functional structure seven rules that correspond to the listed flow combination cases can be defined:

1. IF F9 AND (F7 OR F4) THEN F1
2. IF F1 AND F13 THEN F2
3. IF F2 THEN F3 OR F4
4. IF F3 THEN F5
5. IF F5 AND F8 THEN F6 AND F14
6. IF F6 THEN F7
7. IF F10 AND F11 AND F12 THEN F8

It is quite obvious that the defined rules easily can be implemented in some expert system shell providing reasoning about relationships of components and/ or cause–consequence relationships of the system.

## 5. CONCLUSIONS AND FUTURE WORK

In structural modelling framework significant efforts are made to reduce a workload of human experts during the model building process. For this purpose the methods and algorithms for automatic generation of topologies of structural models have been developed and implemented. The paper describes different cases of logic of input and output flows of system components and defines corresponding transformations. The new results allow extending already developed formal method of transformation of a model of morphological structure into a model of functional structure. For all cases of transformation the corresponding logical expressions given in form of IF … THEN… rules are described. This provides the possibilities to reason and to make decisions about the structure and operation of the system as a whole. The future work is connected with the implementation of the proposed modified algorithm of formal transformations.

## ACKNOWLEDGMENTS

„This work has been partly supported by the European Social Fund within the National Programme „Support for the carrying out doctoral study programm's and post-doctoral researches" project „Support for the development of doctoral studies at Riga Technical University".

## REFERENCES

Abu-Hanna A. and Jansweier W. N. H., 1994. Modelling Domain Knowledge Using Explicit Conceptualization. *IEEE-Expert 9(5)*, pp. 53-64.

Fang M., 1994. MFM Model-Based Diagnosis and Implementation. *Report 94–ID–712*, Institute of Automatic Control systems, Technical University of Denmark, Lungby

Grundspenkis J., 1997. Structural Modelling of Complex Technical Systems in Conditions of Incomplete Information: A Review. In: *Modern Aspects of Management Science, No 1.* Riga, Latvia, pp. 111-135.

Grundspenkis J., 1999. Reasoning Supported by Structural Modelling. In: Intelligent Design, Intelligent Manufacturing and Intelligent Management, (Wang K. and Pranevicius H., eds.), *Kaunas University of Technology Press, Technologija*, pp. 57-100.

Grundspenkis J. and Jekabsons J., 2001. Model Transformation for Knowledge Base Integration within the Framework of Structural Modelling. In: Databases and Information Systems. *Fourth International Baltic Workshop, Baltic DB&IS 2002,* Vilnius, Lithuania, May 1-5, 2000, Selected Papers, (J. Barzdins and A. Caplinskas, Eds.). Kluwer Academic Publishers, Dordrecht, The Netherlands, pp. 261-274.

Grundspenkis J., 2002. Reasoning in Structural Model–Based Diagnosis. In: *Proceedings of 4th International Conference on Quality, Reliability and Maintenance, QRM 2002*, *Oxford,* March 21-22, ( McNulty G.J., ed.), *Professional Engineering Publicity,* London, UK., pp. 295-298.

Grundspenkis J., 2004. Automated Transformation of the Functional Model into the Diagnosis Knowledge Base. In: *Proceedings of the 5th International Conference on Quality, Reliability and Maintenance*, *QRM 2004, Oxford,* 1-2 April, 2004, (McNulty G.J., ed.), *Professional Engineering Publishing*, London, UK, 2004, pp. 295-298.

Harandi M. T. and Lange R., 1990. Model-Based Knowledge Acquisition, In: Adeli, H. ed., *Knowledge Engineering, Vol. 1*, McGraw Hill, New York, pp. 103 – 129.

Kurki M., 1995. Model-Based Fault Diagnosis for Mechatronic Systems. Thesis (PhD), Technical Research Centre of Finland, VTT Publications 223, Espoo.

Tutte W.T., 2001. Graph Theory, Cambridge University Press, New York.

## AUTHORS BIOGRAPHY

**Janis Grundspenkis** is a professor at Riga Technical University, the dean of the Faculty of Computer Science and Information Technology, the director of the Institute of Applied Computed Systems, and the head of the Department of Systems Theory and Design. He received his Dr.sc.ing. degree in 1972, Dr.habil.sc.ing. degree in 1993 both from Riga Technical University. His research interests are agent technologies, knowledge engineering and management, and structural modelling. He is a member of Institute of Electrical and Electronics Engineers (IEEE), and Association for Computing Machinery (ACM). He is a full member of Latvian Academy of Science.

**Ieva Zeltmate** is the system analyst and PhD student at the Department of Systems Theory and Design of Riga Technical University. Her scientific interests are associated with system analysis, design and development of knowledge representation systems and systems for structural modelling.