On Support Functions for the Development of MFM Models

Heussen, Kai; Lind, Morten

Published in:
Proceedings of the first International Symposium on Socially and Technically Symbiotic System

Publication date:
2012

On Support Functions for the Development of MFM Models

Kai Heussen[1], Morten Lind[2]

[1] Department of Electrical Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark
(Tel: +45-4525-3542, E-mail: kh@elektro.dtu.dk)
[2] Department of Electrical Engineering, Technical University of Denmark, Kgs. Lyngby, Denmark
(Tel: +45-4525-3566, E-mail: mli@elektro.dtu.dk)

Abstract: A modeling environment and methodology are necessary to ensure quality and reusability of models in any domain. For MFM in particular, as a tool for modeling complex systems, awareness has been increasing for this need. Introducing the context of modeling support functions, this paper provides a review of MFM applications, and contextualizes the model development with respect to process design and operation knowledge. Developing a perspective for an environment for MFM-oriented model- and application-development a tool-chain is outlined and relevant software functions are discussed. With a perspective on MFM-modeling for existing processes and automation design, modeling stages and corresponding formal model properties are identified. Finally, practically feasible support functions and model-checks to support the model-development are suggested.

Keywords: Multilevel Flow Modeling (MFM), Model Development, Analysis and Design, Software.

1. INTRODUCTION

As a modeling language and as knowledge-base for reasoning applications, MFM has been developed over many years and its potential for applications in several technology domains has been demonstrated. The development of an MFM model as well as MFM applications requires – or enables – the integration of different types of engineering knowledge. This property of MFM, the role as a tool for integration of knowledge, makes MFM both powerful and difficult to apply.

In modeling practice with MFM there is typically a long tool-chain between graphical model development and its final application, e.g. in a HAZOP study. Model-development in itself is a challenging task and both simple syntactic mistakes and more complex semantic mistakes occur in the process. As any modeling process it requires iterations between modeling and application to mature event formally correct model. Here a long tool-chain becomes tedious and may impact the ability of a modeler to focus on essential interpretation tasks. Some of these difficulties, however, can be avoided. As the feedback collected from model-validation, model-building as well as from the failure of certain reasoning tasks is experienced as valuable for the development of high-quality models, similar information could be provided directly in the modeling environment.

In order to motivate the broad scope that a MFM-oriented modeling environment ought to support, this paper first presents a review of MFM in Section 2. The type of knowledge captured in a MFM model and other related domains of knowledge are analyzed in Section 3. Focusing on software tools for the support of MFM modeling and application development, Section 4 presents a global tool-chain perspective, integrating modeling with reasoning, offline and online applications. In Section 5, the modeling process is viewed in a knowledge acquisition perspective and divided into stages, corresponding to formalized levels of knowledge. Finally the realization of support functions in a dedicated MFM modeling environment is discussed in Section 6.

2. REVIEW OF APPLICATIONS AND MODEL REQUIREMENTS

A review of the existing rather comprehensive literature on MFM is not straight forward because the contributions of most papers shown in the list of references are relevant for more than one perspective including modeling concepts and methodology, technology domain and model application. We will therefore categorize the research contributions along these three dimensions in the following.

2.1 Modeling and reasoning methodology

A significant part of the research on MFM has contributed to the development of concepts and modeling methodology. Concepts and methodology has been evolving over several decades and is still in ongoing as part of the application of MFM in different technology domains described below. Contributions to the foundational concepts are presented in [21, 23-26, 28, 30, 32-35, 39]. Contributions to modeling methodology are presented in [3, 15, 34]. Several research contributions address the reasoning capability of MFM. These contributions include [1, 4, 17-20, 22, 31, 38]. The relations between MFM and other modeling approaches such as differential equations have also been investigated [7].

Table 1 Overview of proven MFM applications

Application Category / Technology Domain	Process Analysis & Design	Control Structure A&D	Risk Analysis	Visualization & Decision Support	On-line Diagnosis & Risk Monitoring
Chemical Engineering	[5][50]		[41]	[8]	
Nuclear Power Plants	[34][35][36]		[44]	[9][45]	[12][37][46][47]
Electric Power Systems		[13][14][16][42]			
Other energy conversion	[48]				[15]
General	[6][23][24][26][27][8][30][32][38][39]	[19][29][33]	[31]	[2][7][10][21][40]	[4][17][1][20][22][25][31][43]

Tools for building MFM models have been developed by several research groups. These efforts are not widely published (see however [49]), also partly because they are under development [51].

2.2 Technology domain

MFM has been used to represent a variety of complex dynamic processes including energy conversion systems like fossil power plants [17], nuclear power generation [9, 12, 34-37], gas turbines [48] and ship engines [15]. MFM has also been used to model power transmission and distribution systems [13, 14, 42] and for chemical engineering systems such as oil refineries [8], distillation columns [41] and biochemical processes [15]. Ongoing research at DTU develops MFM extensions for representations of chemical reactions.

MFM has proven to be able to cover these domains and to be robust enough to be able to assimilate extensions required for the different domains. MFM can in this way highlight both generic features and commonalties and differences between domains.

2.2 Model purpose

MFM are used for a variety of purposes within supervision and control of complex automated processes. One group of application includes situation assessment and fault diagnosis for decision support of control room operators. This research includes root cause analysis [20, 22, 25, 31, 38] alarm design [43] and alarm analysis and filtering [17, 18, 37]. MFM is also proposed for on-line risk monitoring [30, 45-47] and for risk analysis of processes in the design phase [41, 44]. Application of MFM for planning of control actions have been investigated by [16, 40]. Recent promising applications of MFM include design of control system architectures [13, 14]. Finally, the role of MFM in design of Human Machine Interfaces has been investigated in [2, 7, 9, 11, 21, 26, 27].

2.3 Overview of research contributions

Table 1 categorizes the research contributions outlined above in a matrix to illustrate the coverage application categories in the different technology domains.

3. MFM IN CONTEXT OF OTHER ENGINEERING KNOWLEDGE

Whereas MFM is focused on the formal expression of goal-function-structure information about a process, it is important for development of MFM models and their use in applications to also understand and, if feasible, formally capture its relation to other domains of knowledge. To develop an overview of these relations, we first identify the knowledge embedded in MFM models, and then develop an overview of other types of knowledge MFM relates to in the engineering process. When viewing the development of MFM models and applications as part of an engineering process, it further becomes relevant to consider embedding and progression of MFM models and applications in the engineering lifecycle.

3.1 Knowledge specification within MFM models

Efficient representation of process knowledge is a prerequisite for knowledge based systems reasoning about complex industrial processes. MFM models are efficient for this purpose because they combine process knowledge on four interdependent levels of specification as shown in Table 2 [31]. These levels are relevant both for the reasoning perspective, as presented in [31], and the modeling phase, as will be seen in Section 5.

The specification of functional knowledge can only be made operational, when its relation to physical and structural specifications can clearly be identified. Such ‘structural’ knowledge is captured in common domain-specific representations, and is not by itself part of MFM.

Table 2 Four levels of knowledge specification embedded in MFM models [31].

Level	Knowledge categories	
4	Event propagation paths	
3	MFM Patterns	
	Influence patterns	Means-end and control patterns
2	Influence relations	Means-end and control relations
1	State dependency relations	

The function-structure, relation, however, can be considered part of the MFM modeling language and has been formulated in detail in [30]. Using an action-role concept with explicit role-entities, also design features such a structural redundancy can be modeled.

3.2 Knowledge embedding in Automation Design

In the engineering process for technical systems, various types of knowledge representation are relevant. Detailed design specifications are, during a system design phase, developed from high-level design goals. Here, repeatedly design choices from one engineering domain have to be translated into requirements for another. Figure 1 illustrates this basic design step and explains the role of design patterns in mediating between requirements and design solutions – the focus is on the structured representation integration

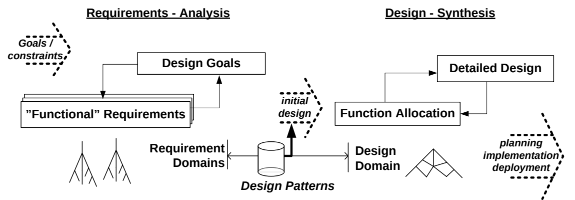

Fig. 1 The step between Requirements Analysis and Design Synthesis can be supported by design patterns, which are patterns in the design domain characterized by a set of properties in the requirements domain [14].

With a perspective on automation design, an embedding of design knowledge representations can be established, from core-process knowledge toward knowledge about enabling technologies:

1. Process knowledge
   - Technology domain representations (e.g. PI-diagrams)
2. Functional requirements knowledge (incl. goal and function knowledge)
   - Threats and Requirements for safety
   - Control structure and behavioral knowledge Performance
3. Instrumentation knowledge (e.g. measurement and actuation)
4. Communication and computation technologies

Each of these layers thus derives *requirements* from the layer above, and its functions are *enabled* or potentially enabled by the layers below. For example, given a process structure, objectives and requirements are specified and have to be translated to a control strategy with detailed control objectives, loop pairing, etc. A fully specified control system would then in turn pose requirements to instrumentation, communication and computation, etc.

The functional representations offered by MFM can be related to layers 2 and 3, where process analysis, control structure design and diagnosis are specific application considerations. Note that the layering of such knowledge does not necessarily imply a rigid sequence. For example, an opportunity-driven approach would be focused on the impact of enabling technologies.

Note that process fundamental knowledge such as knowledge about physics, chemistry, etc. is general *phenomena* knowledge and cannot be categorized meaningfully into a means-ends hierarchy.

3.3 Engineering Lifecycle

The engineering lifecycle considers the overall product development cycle from initial requirements to the deployment and disposal of a process. Stages of an engineering lifecycle have been described in the so-called Waterfall Model:

- Definition Study/Analysis
- Basic Design
- Technical Design/Detailed Design
- Construction
- Testing
- Integration
- Management and Maintenance.

MFM applications outlined above seem to relate particular into the earlier and later phases. The actual work process in systems engineering relates to the specification and analysis phase (see Fig. 2).

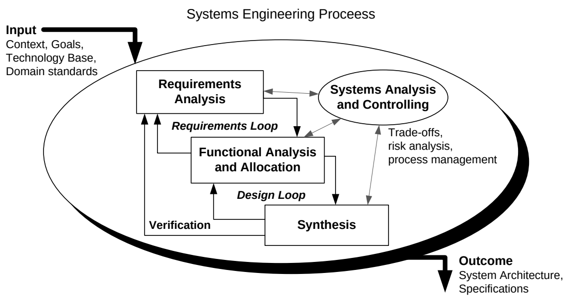
Systems Engineering Proceess
Input
Context, Goals, Technology Base, Domain standards
Requirements Analysis
Systems Analysis and Controlling
Requirements Loop
Trade-offs, risk analysis, process management
Functional Analysis and Allocation
Design Loop
Verification
Synthesis
Outcome
System Architecture, Specifications

Fig. 2 Systems Engineering. Adapted from [14]

Again, the formal structure of the waterfall model describes sequential stages, in Rapid Application Development, it has proven practical to organize the actual work process in a more iterative and concurrent fashion. Simplifications, shortcuts and rapid iterations are often feasible, e.g. enabled by design patterns and experience-based approaches.

Here, prior knowledge and rapid prototyping approaches are employed to more quickly generate useful feedback for the designer.

In this design phase, the main role of MFM models is in support of process analysis, control structure analysis and design, as well as risk analysis.

During system operation and for system maintenance, the MFM-related functions are in support of visualization, decision support, online diagnosis and risk monitoring.

4. GENERIC MFM TOOLCHAIN

The two key operations in working with MFM are a) *modeling and knowledge representation* and b) *reasoning operations* on a given model. In this paper we assume that each of these operations requires its own development platform.

As there are a number of applications for MFM that have in common the need for reasoning about the model a dedicated development environment for MFM application development has been developed at DTU, called MFM Workbench. The key idea is here to provide a platform in which rule-based programming for various MFM applications can be developed. The MFM Workbench can thus be considered a prototyping environment.

A pure MFM environment helps the focused development, but as it has been seen in the discussions above, most applications of MFM require the integration with other knowledge domains on the one hand, and with other tools and data sources on the other.

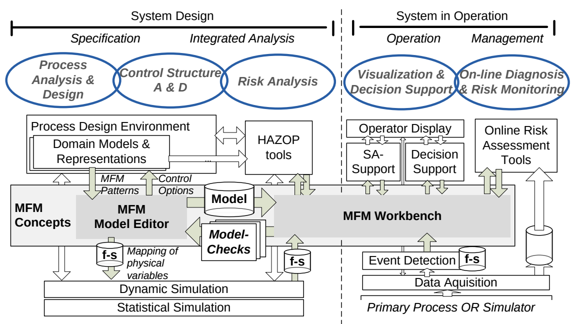

Fig. 3 Overview of toolchain in relation to MFM applications. MFM tools can be relevant during in system specification, analysis and operation phases. Fat text indicates core MFM tools and grey arrows indicate a MFM-oriented interface. The database symbols with **f-s** on them indicate a need for integrating function-structure knowledge at the respective interface.

Fig. 3 presents an overview of the MFM model editor and MFM workbench in context of the above discussed MFM applications in the mapped out into different phases of the MFM lifecycle. The objective of presenting this figure is to describe the purpose and function of MFM oriented tools in relation to other specialized application tools. In each specific application, MFM oriented tools only support the primary application objectives. It becomes clear that tools focused on working with MFM would have to satisfy a range of requirements and offer a number of interfaces for dealing with specific applications.

The MFM-oriented tools are composed of three primary functions:

- Development and maintenance of MFM concepts
- MFM model development
- Prototyping and deployment of MFM-based reasoning functions

The primary development tool for MFM at DTU has been the MFM Workbench. The MFM concept development as well as model development used to be supported by MS Visio®. Whereas the graphical support functions of MS Visio have been powerful for concept development, the weak object orientation and scripting in this environment has increasingly been perceived as a barrier. In a current collaboration between DTU and IFE Halden, a dedicated modeling tool, the MFM Editor, is being developed, building on IFE Halden experience with modeling tools for a variety of purposes [51]. At the same time, the MFM Workbench is being re-engineered to support a more robust prototyping of reasoning modules and interfacing with external applications. Both MFM model editor and MFM Workbench are based on JAVA, which facilitates a seamless integration. In a further step MFM concepts shall be extracted from their separate implementation in each tool toward a more appropriate ontology representation in order to facilitate maintenance and development of a MFM concept ontology. Note also that MFM workbench is presented in within both design phase and operation. Each of these contexts presents very different requirements, but it is believed that an appropriate software design may enable a common prototyping platform for either application category. For real-time applications, a specialized deployment architecture might have to be developed.

For the remainder of this paper the focus will be on the need and feasibility of modeling support functions which would be integrated as assistance modules in the MFM model editor. Its focus is on supporting the human modeler in the graphical model development, to store and retrieve models as well as to provide meaningful support functions.

In particular, it is anticipated that with the presented architecture, modeling support functions can be developed as a side-gain from the development of reasoning tools for other applications. For this to be enabled, a specialized interface between the Model Editor and Workbench shall be developed (refer to Fig. 3: the box ‘Model Checks’).

A support function anticipated in the figure might require another type of software interface. It is related to the mapping between technology domain knowledge representations) and MFM patterns (to be discussed in the following section).

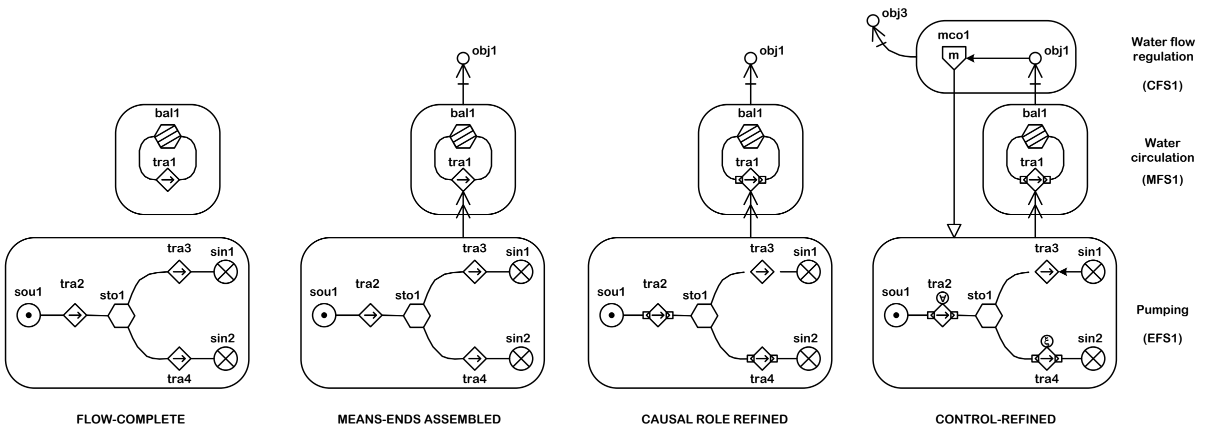

Fig. 5 Exemplification of formally distinct stages of MFM model development.

5. MODEL DEVELOPMENT STAGES

Modeling is encoding of knowledge. As part of it is creative interpretation, there is no principled straightforward path from the modeling intention to an application-ready model. However, as in any other work process, specific modeling tasks can be identified and related to a modeling process that outlines the path from first steps toward a formally correct and meaningful model.

Figure 4 presents an outline of model development stages in relation to the level of the embedded knowledge as discussed in Section 3. The stages are thought from the perspective of modeling an existing process, and arrows indicate meaningful iterations through the stages.

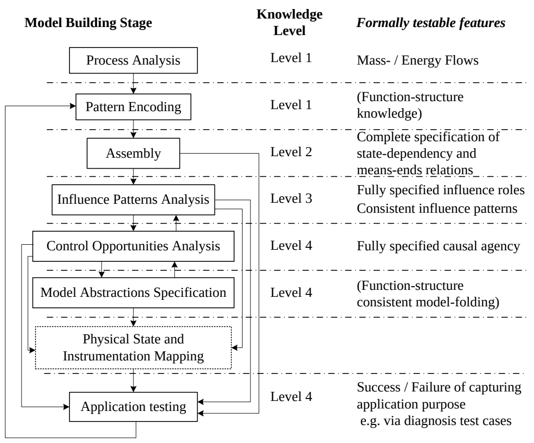
Model Building Stage
Knowledge Level
Formally testable features
Process Analysis
Level 1
Mass- / Energy Flows
Pattern Encoding
Level 1
(Function-structure knowledge)
Assembly
Level 2
Complete specification of state-dependency and means-ends relations
Influence Patterns Analysis
Level 3
Fully specified influence roles
Consistent influence patterns
Control Opportunities Analysis
Level 4
Fully specified causal agency
Model Abstractions Specification
Level 4
(Function-structure consistent model-folding)
Physical State and Instrumentation Mapping
Application testing
Level 4
Success / Failure of capturing application purpose
e.g. via diagnosis test cases

Fig. 4 Model development stages in relation to knowledge level and formally testable features.

It has been found that each of these stages can be related to formal properties of the model which can be checked in a rather straightforward fashion if the graphical partial model is translated to its formal representation.

The concept of the modeling stages is based on the idea that knowledge is added incrementally and iteratively to an MFM model. Whereas the modeling process might not develop homogeneously for the complete model, a certain level of completion is at least locally always the basis for a following stage. These stages of completion also relate to a specific level of process understanding that has been achieved and captured in the model. Note that Fig. 4 also includes the modeling of relations to the process under study (noted as function-structure knowledge), which has been discussed in [30]. First the formal properties of the knowledge levels will be discussed, and then the relation of MFM models to process domain knowledge and MFM applications will be addressed.

5. 1 Formal Interpretation of Stages

Fig. 5 illustrates some of these different stages on a simple MFM model. The MFM model represents a water-circulation process powered by a pump. The stages are presented in a pure form which is meant to convey the formal (and incremental) properties associated with these stages:

- *Process Analysis (Level 1)*: identify mass-flow and energy-flow processes and connected abstract functions.
  Formal completion of this stage is achieved independently for each flow-structure when all functions within a flow structure are connected syntactically correct (FLOW-COMPLETE).
- *Means-Ends Assembly (Level 2)*: identify objectives and goal-directed interactions between flow-structures.
  Formal completion: All flow-structures are interconnected and objectives and main-functions have been identified at all levels (MEANS-ENDS ASSEMBLED).
- *Influence Pattern Analysis (Level 3)*: identify influence relations [38] for all connections and ensure consistency of influence patterns.
  Formal compl.: All function connections have been typed into either participant or influencer; the patterns are formally consistent [13] (CAUSAL ROLE REFINED).
- *Control Opportunities Analysis (Level 4)*: identify those functions whose state is subject

to external influence, such as actuation, disturbance or (adjustable) fixation; assign control functions to objectives and actuation.
Formal compl.: At least all influencer-free transports have been assigned an external agent-role; all control functions have a specified actuation path (event propagation to objective) [13] (CONTROLREFINED).

This concludes the formal stages illustrated in Fig. 5, and also the points for which straightforward model-checks can be defined at present. However, as indicated in Fig. 4, further stages can be identified and supported:

- *Model Abstraction Specification:* Control encapsulation and other types of abstraction-effects relevant for the application are identified and are supported by meaningful and consistent function-structure relations.
- *Physical State and Instrumentation Mapping*: This stage is meant to prepare the deployment of a model for respective application purposes. During model development it is not a chief concern to relate measurements and other observable states to MFM functions. For any concrete application in operation, however, the definition of both the concrete physical variables to be represented by a function as well as the bounds to capture abnormal function-states is essential. Failure in identification of these mappings could result from a modeling mistake in a previous stage.
- *Application testing*: This stage is an integral part of the model maturing process. For any application, test-cases should be defined. The failure of a model to support a test case is a strong indicator for an incomplete, inaccurate, or in another way inappropriate model.

The above stages and model checks include straightforward syntactic checks, but also checks for semantic consistency, such as tests for influence patterns and event propagation-

5.2 From Technology Domain to MFM Model

The essential creative step in modeling is the interpretation of given technology domain knowledge in terms of MFM functions and relations.

The intuitive approach for creating a new model is to start with an energy/mass-flow analysis and to simply do the modeling from scratch, function-by--function, relation by relation. More systematic approaches have been proposed in [24], [3] and [15].

If other models in the same domain exist, it can be helpful to begin on given model and to work by modification – an approach that would save time, but might reduce the necessary attention to detail.

Yet, it has been found that –for a given technology domain – certain patterns of MFM functions can be found to re-occur, consistently with patterns in the technology domain [34,13]. This observation has led to the idea of a pattern-based model-creation approach.

In a simple and only mildly formal approach, a domain-specific *pattern library* can be build. The library can be extended with every new model created. Here it would be essential to co-document the related technology domain feature that has been modeled, in the sense of Design Patterns discussed in Section 3.1.

In a more formal and automated approach, the same idea can be applied if a structured formal representation of the technology domain is available and a strict pattern-mapping has been established [13]. It has been demonstrated that – in the case of electric power systems – a completely formalized MFM model could be created by execution of a script, thus enabling also the generation of large-scale models. Further research is required to identify whether such an approach is feasible also in other technology domains.

5.3 How do Stages Support the Modeling Process?

The formal stages identified above are, analog to the engineering lifecycle discussed in Section 3.3, characterized by sequential dependency, but equally allow alternative pathways and overlapping stages within the same modeling task. The interpretation of these stages is therefore not thought of as a formal sequence of modeling steps, but rather as ‘backlog’ estimation and ‘debugging’ features.

If for example a subset of a model can be identified as rather complete, whereas other parts would be identified as ‘behind’ in the modeling process, this information would support the modeler in focusing attention: It could for example be preferred to finalize a certain subset of the model first for prototyping purposes.

The most practical contribution would however come from a minimized debugging effort which currently involves iterations through the complete application chain.

A further aspect can be related to a classification and navigation of stored models for application purposes. For example, not all applications require a model completion to Level 4, and it is likely that MFM patterns in a pattern library include also Level 1 and Level 2 patterns.

In summa, the modeling stages support a lifecycle perspective for MFM models. It is clear that the creative aspect, the striving for a meaningful and relevant model cannot be achieved by formal checks alone – but a well-supported modeling process leaves attention for meaningful reflections, and rapid iterations which will lead faster to more mature models.

6. SUPPORT FUNCTIONS

The general outline of meaningful support opportunities related to MFM modeling stages has been presented in the previous section. Here we shall focus on the interaction between tool and modeler.

6.1 Build-in and Elementary Support Functions

Elementary support functions are those build into the model editor as modes of interaction with the modeling canvas. Some that have been implemented in the MFM Editor/ShapeShifter [51] framework include:

- Model-extension: Offering only syntactically correct functions
- Automatic numbering / naming
- Annotations to MFM entities
- Marking of connection points indicating connectivity, etc
- Switching relations connectivity view: main-function/ target-function connectivity for means-function relations can switch connection point between flow-structure and function, improving overview in larger models.

Further functions considered desirable include:

- Offering function replacement for syntactically equivalent functions (balance – storage; transport – bi-transport, etc.
- Definition of multiple views on a model.

This list is of course non-exhaustive and is only meant to illustrate types of interactions, and it showed that a good interaction between tool developer and users provide a foundation for fruitful tool improvements.

6.2 Implementation of Model-Checks

The modeling stages as discussed in the previous section formalize properties of an MFM model that can easily be tested. Such a test would be developed in the MFM Workbench – in most cases by simply creating additional output from already existing reasoning functions.

Once the respective model-check has been implemented, it can be made available through a dedicated interface between MFM Editor and MFM Workbench. The MFM Editor offers available Model Checks as executable plug-ins to the Model-Developer. Execution of a Model Check will then provide state (e.g. success/failure) information about the MFM entities in the respective model regarding its Model Stage.

The information returned can then be utilized by the MFM Editor, for example to provide graphical feedback about the result. Depending on the check, the result can be indicated e.g. by color-coding the set of functions that passed the check or additional annotations.

Beyond this detailed information about individual patterns, the check also results in information qualifying the entire model, which can be stored and displayed in the MFM Editor as state information associated with the model file.

6.3 Implementation of Pattern Library

A pattern library as described above would require an association of technology domain data with corresponding MFM patterns. A searchable pattern library, structured by technology domains would be a desirable feature. As the function-structure relation modeling has not been included at this point, this full library feature is not practical yet.

Instead, the possibility of selecting a partial MFM model and defining a collection of partial MFM models has been implemented in a practical fashion.

Automatic model translation as discussed in Section 5.2 is beyond the scope of this paper. It is expected that a robust implementation of this type of translation will require advanced semantic technologies, and also further development of respective domain ontologies in the technology domains. In Power Systems the domain ontologies for data exchange are relatively advanced, consider e.g. the ‘Common Information Model’ (CIM) for Power Systems, IEC 61970 and related standards. For example in chemical engineering, standardization efforts are under way but prove difficult (consider e.g. efforts toward ISO 15926).

6.4 Model Lifecycle Support

In general, it is found that an integrated, project-oriented approach to modeling and reasoning with MFM models is favorable.

So far, a template structure for different classes of MFM related file types has been created and also the MFM editor is structured into modeling projects rather than single files.

Further lifecycle support can be achieved with a model-maturity tracking system, which can be interpreted as meta-information generated for example on the basis of the above-discussed model-checks. The meta-data would for example enable a reasoning system to identify the validity of a model for a requested reasoning task. Consider keywords classifying the states such as:

- RAW / DRAFTING
- PARTIALLY FLOW-COMPLETE
- BUILDABLE
- CAUSAL ROLE CONSISTENT
- FUNCTION-STRUCTURE COMPLETE

Beyond these formal properties, further information can be included that would inform about the maturity of a model based on application experience. Whether such knowledge can be formalized in a meaningful fashion is not clear at this time. For example successful (and failed) model-applications could however be recorded here.

A last essential item of meta-information is the version of the MFM concepts that have been employed in the model development. MFM has been and is a work in progress and improved modeling concepts will continue to extend and improve potential applications. It is therefore essential for the model lifecycle for any larger application to keep track not only of the model versions but also on the version of MFM concepts it has been developed in.

7. CONCLUSION

This paper presented an introduction and overview to the MFM application context in a broad scope. Not the MFM modeling itself has been in focus, but rather the applications and contextual requirements of the modeling process as knowledge acquisition. The presentation of an MFM-oriented tool chain picture also expresses the interface requirements that will have to be addressed in further work.

It was found that formal checks can support the development of formally complete and consistent models. Certainly the clarification of representation problems or modeling mistakes can not directly be assessed by such formal checks. However, as the iterations required for modeling become faster, more contextual information is available, and correctness checks are automatic, a resulting more focused reflection about modeling problems is still expected to help improving model quality in general.

Further, the introduction of life-cycle related model information and other meta-information should simplify the handling of several model versions and their relation to different version of the MFM concepts.

In further work on tool development, supporting functions for the MFM concept development should be addressed. Knowledge representation can be considered its own application category – to be supported by relevant semantic technologies.

The discussion on shared interface definitions could also be interesting in relation the research and development of MFM applications together with other, special purpose or technology specific application software.

8. ACKNOWLEDGEMENT

The present work is part of an ongoing cooperation between the OECD IFE Halden project and DTU on the use of MFM for innovative monitoring systems. The authors acknowledge the contributions of H. Thunem to specifications of the MFM Editor discussed here.

REFERENCES

[1] F. Dahlstrand. Consequence Analysis Theory for Alarm Analysis. Knowledge Based Systems, 15(1), 2002: 27-36.

[2] K. Duncan and N. Prætorius. Flow Displays representing Complex Plant for Diagnosis and process Control. Prof. Cognitive Science Approaches to Process Control (CSAPC), Siena Italy, October 24-27 1989.

[3] M. Fang. MFM Modeling Method and Application. Technical Report 94-D-713, Department of Automation, Technical University of Denmark, March 1994.

[4] M. Fang and M. Lind. Model-Based Reasoning Using MFM. Proceedings. Pacific Asian Conference on Expert Systems (PACES). Huangshan China, 1995.

[5] K. V. Gernaey, M. Lind and S. B. Jørgensen, "Towards Understanding the Role and Function of Regulatory Networks in Microorganisms". In: L. Puigjaner and G. Heyn (Eds), Computer Aided Process & Product Engineering, Wiley-VCH, Weinheim Germany, 2004.

[6] A. Gofuku, Y. Seki and Y. Tanaka. Representation of Goal-Function-Structures Information for Efficient Design of Engineering Systems. Proc. Int. Symposium. Cognitive Systems Engineering in Process Control (CSEPC), Kyoto Japan, November 12-15 1996.

[7] A. Gofuku and Y. Tanaka, "A Combination of Qualitative Reasoning and Numerical Simulation to Support Operator Decisions in Anomalous situations, In: Proc. 3´rd IJCAI Workshop on Engineering Problems for Qualitative Reasoning, p. 19-27, Aug 23-29, 1997.

[8] A. Gofuku and Y. Tanaka. Application of Derivation Technique of Possible Counter Actions to an Oil Refinery Plant, Proc.4´th IJCAI Workshop on Engineering Problems for Qualitative Reasoning, Stockholm Sweden, July 31-August 6, 1999: 77-83.

[9] A. Gofuku, Y Ozaki and K. Ito, "A Dynamic Operation Permission System for Pressurized Water Reactor Plants", In: Proc. ISOFIC´2002 International Symposium on the Future of I&C for NPP, p. 360-365, Seoul Korea, Nov. 7-8, 2002.

[10] A. Gofuku, T. Ohi and K. Ito, "Qualitative Reasoning of the Effects of a Counteraction Based on a Functional Model, Proc. CSEPC´2004, Sendai Japan, November 4-5, 2004.

[11] A. Gofuku. Support systems of plant operators and designers by function-based inference techniques based on MFM models. Int. J. Nuclear Safety and Simulation, 2011, 2(4).

[12] G. Gola, M. Lind, H. Thunem, A. Thunem, E. Wingstedt and D. Roverso. Multilevel Flow Modeling for Nuclear Power Plant Diagnosis. Proc. European Safety and Reliability Conference, ESREL2011,Troyes France, Sept. 18-22, 2011.

[13] K. Heussen, A. Saleem and M. Lind. Control Architecture of Power Systems: Modeling of Purpose and Function, Proceedings IEEE PES General Meeting, Calgary Canada, July 26-30 2009.

[14] K. Heussen. Control Architecture Modeling for Future Power Systems. PhD Thesis, Technical University of Denmark, Nov. 2011.

[15] S. S. Jørgensen. Fault Diagnosis Using Generic Multilevel Flow Modelling Models: In Theory and Praxis. PhD dissertation, Department of Automation, Technical University of Denmark, 1993.

[16] M. N. Larsen. Deriving Action Sequences for start-up using Multilevel Flow Models. PhD dissertation, Department of Automation, Technical University of Denmark, 1993.

[17] J. E. Larsson. Diagnostic Reasoning Strategies for Means-End Models, Automatica, 1994.

[18] J. E. Larsson, "Diagnosis Based on Explicit Means-end Models", Artificial Intelligence, 80(1), 29-93, 1996.

[19] Lind, M. The Use of Flow Models for Design of Plant operating procedures. Proc. IWG/NPPCI Specialists Meeting on Procedures and Systems for Assisting an Operator during Normal and Anomalous Nuclear Power Plant Operating Conditions, Garching Germany, December 5-7, 1979.

[20] M. Lind. The Use of Flow Models for Automated Plant Diagnosis, In: J. Rasmussen and W. B. Rouse: Human Detection and Diagnosis of System Failures, Plenum Publishing Corporation, 1981, 411-432.

[21] M. Lind and J. Rasmussen. Coping with Complexity, European Conference on Human Decision and Manual Control, Delft The Netherlands, 1981.

[22] M. Lind. Diagnosis Using Multilevel Flow Models: Diagnostic Strategies for the P96 Demonstrator. EU-ESPRIT P96, July 1988.

[23] M. Lind. Representing Goals and Functions of Complex Systems, Technical University of Denmark: Department of Automation, 1990.

[24] M. Lind. Modeling Goals and Functions of Complex Industrial Plant, Journal of Applied Artificial Intelligence, 1994, 8:259-283.

[25] M. Lind, "Interpretation Problems in Modeling Complex Artifacts for Diagnosis", In: Proc. Cognitive Engineering for Process Control (CSEPC´96), Kyoto Japan, Nov. 12-15, 1996.

[26] M. Lind. Plant Modeling for Human Supervisory Control, Trans. Inst. Measurement and Control, 1999, 21(4/5):171-180.

[27] M. Lind, "Making Sense of the Abstraction Hierarchy in the Power Plant Domain", Cognition Technology and Work, 5(2), p.67-81, 2003.

[28] M. Lind. The why, what and how of functional modeling. Proc. Int. Symp. on Symbiotic Nuclear Power Systems for the 21´th Century (ISSNP), Tsuruga Japan, 2007.

[29] M. Lind. A Goal Function Approach to Analysis of Control Situations, Proc. 11´th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Human-Machine Systems, Valenciennes France, 2010.

[30] M. Lind. Knowledge Representation for Integrated Plant Operation and Maintenance, Proc. 7´th ANS International Topic Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies NIC&HMIT, Las Vegas, Nevada, November 7-11, 2010.

[31] M. Lind. Reasoning about Causes and Consequences in Multilevel Flow Modeling. Proc. European Safety and Reliability Conference, ESREL 2011,Troyes France, September 18-22, 2011.

[32] M. Lind. An Introduction to Multilevel Flow Modeling. International Journal of Nuclear Safety and Simulation, 2011, 2(1), 22-32.

[33] M. Lind. Control functions in MFM: basic principles. International Journal of Nuclear Safety and Simulation, 2011, 2, June 2011.

[34] M. Lind, H. Yoshikawa, S. B. Jørgensen, M. Yang, K. Tamayama, K. Okusa. Multilevel flow modeling of Monju Nuclear Power Plant. International Journal of Nuclear Safety and Simulation, 2011, 2(3).

[35] M. Lind, H. Yoshikawa, S. B. Jørgensen, M. Yang, K. Tamayama and K. Okusa. Modeling Operating Modes for the Monju Nuclear Power Plant. Proc. 8´th ANS International Topic Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies NIC&HMIT, San Diego, California, July, 2012.

[36] J. Liu, H. Yoshikawa and Y. Zhou. Application of Multilevel Flow Modeling to describe complex processes in a nuclear fuel cycle. Proceedings of Cognitive Systems Engineering in Process Control (CSEPC2004), Sendai Japan, November 4-5, pp. 114-120, 2004.

[37] J. Ouyang, M. Yang, H. Yoshikawa, Y. Zhou and J. Liu. Alarm Analysis and Supervisory Control of PWR Plant. In Proceedings of Cognitive Systems Engineering in Process Control (CSEPC 2004). Sendai, Japan, November 4-5, pp. 61-68, 2004.

[38] J. Petersen. Causal Reasoning Based on MFM, Proceedings International Symposium on Cognitive Systems Engineering in Process Control (CSEPC), Taejon Korea, November 22-25, 2000.

[39] J. Petersen. Countermeasures and Barriers. Proc. Annual Conference of European Association of Cognitive Ergonomics (EACE´05), Chania Greece, September 29-October 1, 2005: 43-50.

[40] E. Souza and M. Veloso. AI Planning in Supervisory Control Systems, Proc. IEEE Int. Conf. Systems, Man and Cybernetics, Beijing China, October 14-15 1996: 31353-3158.

[41] N. L. Rossing, M. Lind, N. Jensen and S. B: Jørgensen. A Functional Hazop Methodol ogy, Computers in Chemical Engineering, 2 010, 34(2): 244-253.

[42] A. Saleem and M. Lind. Reasoning about

Control Situations in Power Systems, Proc. 15´th International Conference on Intelligent System Applications to Power Systems (ISAP), Curitiba Brazil, November 8-12, 2009.

[43] T. Us, N. Jensen, M. Lind and S. B. Jørgensen. Principles of Alarm Design, International Journal of Nuclear Safety and Simulation, 2011, 2(1).

[44] M. Yang, Z. Zhang, M. Peng and S. Yan. Modeling Nuclear Power Plant with Multilevel Flow Models and Its Application in Reliability Analysis, Proc. Int. Symp. on Symbiotic Nuclear Power Systems for the 21´th Century (ISSNP), Tsuruga Japan, 2007.

[45] M.Yang, Z. Zhang, H. Yoshikawa, M. Lind, K. Ito, K. Tamayama, and K. Okusa. Integrated Method for Constructing Knowledge Base System for Proactive Trouble Prevention of Nuclear Power Plant. International Journal of Nuclear Safety and Simulation, 2(2), pp.140-150(2011).

[46] H. Yoshikawa, M. Yang, M. Hashim, M. Lind, and Z. Zhang, "Design of Risk Monitor for Nuclear Reactor Plants", International Journal of Nuclear Safety and Simulation, 2(3), pp.265-273, 2011.

[47] H. Yoshikawa, M. Lind, M. Yang, M. Hashim and Z. Zhang. Configuration of Risk Monitor System by Plant Defense-In Depth Monitor and Relibility Monitor. Proc. 8´th ANS International Topic Meeting on Nuclear Plant Instrumentation, Control and Human-Machine Interface Technologies NIC&HMIT, San Diego, California, July, 2012.

[48] Y. Zhou, H. Yoshikawa, W. Wu, M. Yang and H. Ishii. Modeling Goals and Functions of Micro Gas Turbine System by Multilevel Flow Modeling. The Transactions of Human Interface Society of Japan, 6(1), 59-68, 2004.

[49] Y. Zhou, H. Yoshikawa, M. Yang, J. Ouyang, J. Liu and W. Wu. MFMS A graphical Interface Tool for Mulitlevel Flow Modeling. Proc. 9'th IFAC/IFIP/IFORS/IEA Symposium on Analysis, Design and Evaluation of Human-Machine Systems. Atlanta USA, September 7-9 2004.

[50] I. López-Arévalo, R. Bañares-Alcántara, A. Aldea, A. Rodigues-Martinez, L. Jiménez. Generation of process alternatives using abdtract models and case-based reasoning. Computers % Chemical Engineering. (31)18, 15 August 2007, pp 902-918.

[51] H. Thunem, A. Thunes and M. Lind. Using and Agent-Oriented Framework for Supervision, Doagnosis and Prognisis Application in Advanced Automation Environments. Proc. European Safety and Reliability Conference, ESREL2011,Troyes France, Sept. 18-22, 2011.