Safety and Reliability – Safe Societies in a Changing World – Haugen et al. (Eds)
ISBN 978-0-8153-8682-7

Current status of the MFM suite for diagnostic and prognostic reasoning of industrial process plants

Harald P.-J. Thunem
Institute for Energy Technology, OECD Halden Reactor Project, Norway

ABSTRACT: This paper presents the status of a software system, the Multilevel Flow Modeling (MFM) Suite, dedicated to the design and analysis of MFM models related to diagnostic and prognostic analysis of physical processes. New and updated features of the system are described, as well as some examples of its practical use. The paper also briefly describes how the system facilitates the collaboration between control room and field operators via the Android-based MFM Viewer app.

1 INTRODUCTION

Multilevel Flow Modeling (MFM) (Lind, 2011) is a methodology for graphical modeling of industrial processes by representing the goals and functions of industrial plants. The purpose is to model the combined functions of any number of physical process components, which together provide the means to achieve one or more goals. The model may then be used for diagnostic and prognostic purposes to determine the possible causes and potential consequences of unwanted process events.

Since MFM models consist of graphical, interconnected elements arranged in specific structures, there is an apparent need for dedicated software to design them. The MFM models also need to be connected to the process, the functionality of which they represent.

For several years such a dedicated software system, the *MFM Suite*, has been under development at the Institute for Energy Technology (Thunem, 2013, Thunem and Zhang, 2015). The system will allow a user to graphically design and verify the semantic correctness of MFM models, in addition to creating graphical models of the industrial process. It will provide associations between process components and MFM functions. Using a dedicated MFM reasoning engine developed at the Technical University of Denmark (DTU), the system will perform diagnostic and prognostic analyses of anomalous events on online process data.

This paper provides an update of the system's features, a brief description of results from applying it in a practical experiment, and the functionality and use of the Android-based MFM Viewer app.

2 APPLICATIONS IN THE MFM SUITE

The MFM Suite primarily consists of three applications: the *MFM Editor* for graphical creation, editing and verification of MFM and process models, the *MFM Runtime* for real-time acquisition and analysis of on-line sensor data, and the *MFM Playback* for step-wise and simulated real-time playback and analysis of sensor data. Since the applications share a lot of functionality, they are all based on the *MFMApplication* Java class.

The MFM Suite also includes a simple launcher from which to start the applications.

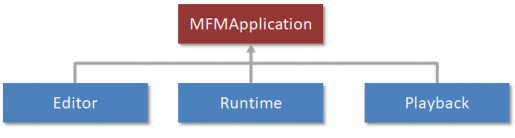

Figure 1. Inherited functionality in MFM Suite applications.

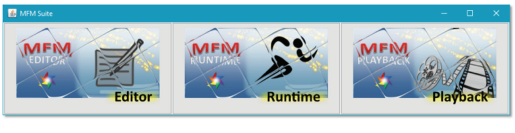

Figure 2. The launcher application.

3 THE EDITOR APPLICATION

The *MFM Editor* contains two essential graphical modelers, the MFM modeler and the process modeler. As both are based on the *ShapeShifter* framework (Thunem et al, 2011, Thunem, 2012), they share a lot of functionality.

3.1 *Meta-model display*

Most complex processes may be in more than one operating state (e.g. start-up, normal, shutdown), and the functionality of the process components and the process goals may depend on the operating state. Therefore, it may be necessary to create separate MFM models to describe the component functionalities and goals for each of the operating states.

The MFM Suite applications includes a meta-level display (Figure 3), which allows MFM models to be connected by arrows, indicating transitions from one operating state to another.

Note that only *one* MFM model may be considered *active* at any point, and any MFM reasoning analysis will be performed on the active model only. This furthermore requires that for each MFM model, the alarm limits for process sensors associated with MFM functions must be set individually, as their alarm limits may depend on the operating state.

3.2 *Click-and-drop model design*

Both the MFM and the process modelers provide graphical click-and-drop design of models in a manner similar to e.g. Powerpoint. For both modelers, a list of available components is shown to the left of the editing area (see Figure 4 and Figure 5). Clicking on one of them changes the cursor into a miniaturized version of the component. Clicking anywhere within the editing area will place the component at this point.

The MFM modeler provides another, faster method which also facilitates the creation of semantically correct models. When selecting an MFM function in the editing area, a list of allowed (according to MFM rules) connections (consisting of one MFM relation and one MFM function) will be displayed (see Figure 6). Clicking on one of them will create the connection from the selected function and place it at a suitable position. The modeler will automatically select the newly added function and update the list of available connections. In this way, complex MFM structures can be designed in relatively few steps.

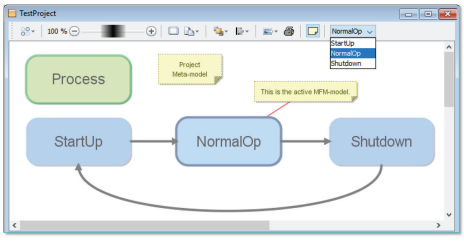

Figure 3. A project's meta-model.

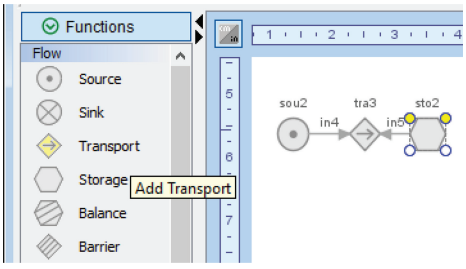

Figure 4. Designing an MFM model by click-and-drop.

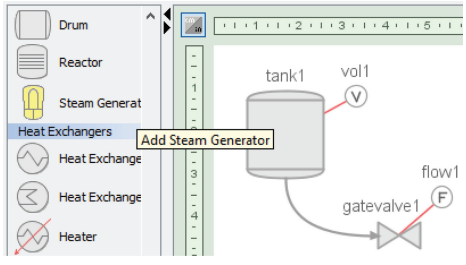

Figure 5. Designing a process model by click-and-drop.

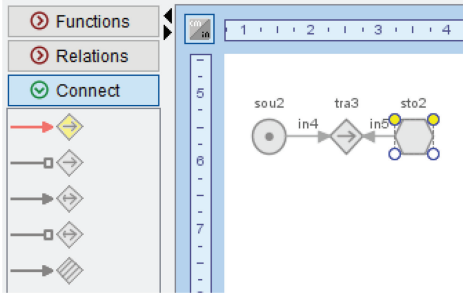

Figure 6. Assisted design of MFM structures.

4 THE RUNTIME AND PLAYBACK APPLICATIONS

The *MFM Runtime* and *MFM Playback* applications use the same *ShapeShifter*-based graphical components for displaying the MFM and process models as the MFM Editor, however in this case the editing functionality is disabled.

4.1 *Separate analysis threads*

The applications will perform diagnostic and prognostic analyses on a given MFM model using the reasoning rules that are explained in detail in (Zhang, 2015). The analysis processes will run in separate execution threads, which will set flags to avoid e.g. starting a new diagnosis before the previous has terminated (however, a diagnosis may run concurrently with a prognosis). This will prevent two problems; the analysis process will not lag the sensor sampling in cases where the sampling interval is shorter than the analysis time, and the user interface remains responsive since the main application thread is not blocked (Figure 7).

4.2 *Sensor display based on value and origin*

In addition to indicating the sensor range by changing the shape, color coding is used to indicate whether the sensor's value is obtained from the process, manually set by an operator, or inferred from an MFM model analysis.

An example is shown in Figure 8, where the shape (down-arrow) and color (red) of sensor PI444 indicates that it has a value below the low limit, and that the source is the actual process/simulator. This value, possibly along with other abnormal sensor values, has triggered an analysis in the associated MFM model.

The analysis has concluded that a *possible* cause of PI444's low value is a high value in sensor TI463, so this sensor is visualized with an up-arrow shape and a yellow color (indicating that the sensor state is inferred from an MFM analysis). Note that if the sensor is functioning correctly and shows a high value, it would be visualized using the red color. Since it is not, we can conclude that either this is not the cause of the abnormal value in PI444, or the sensor is malfunctioning.

The shape and color of sensor TI465 indicates that the sensor has a normal value/state (round shape), and that it has been set manually by an operator (green color), perhaps as a result of a visual inspection.

4.3 *Sensor trend display*

The process viewer component in the Runtime and Playback applications will display trend graphs for

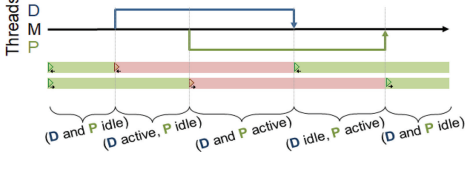

Figure 7. Concurrent main (M), diagnosis (D) and prognosis (P) threads.

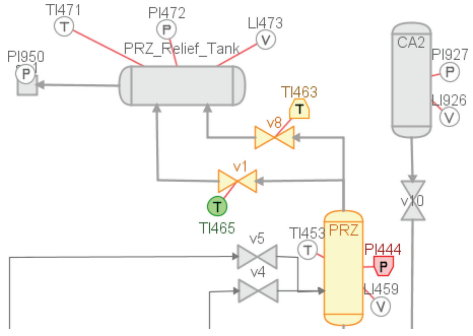

Figure 8. Sensor visualization.

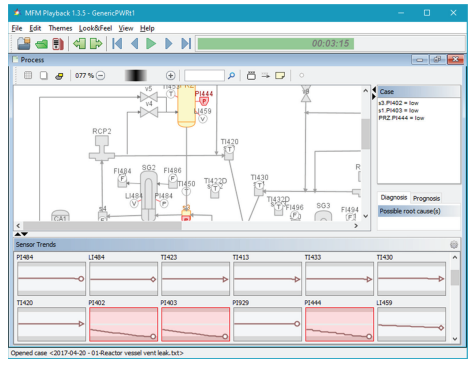

Figure 9. Process sensor trend panels.

selected process sensors (Figure 9). Through a dedicated settings menu the user may select which sensor trends to display (using a searchable selection panel), whether to show different markers depending on sensor type, and specify all colors used in the trends.

4.4 *Playback of existing data (Playback)*

The MFM Playback application will read stored sensor data from files to allow step-wise playback and analysis of previous experiments. The application also includes a slider to move back and forth freely. Furthermore, it can also replay a previous experiment in simulated real-time by activating a timer in a separate thread, and using the stored sampling intervals to update all sensors and re-run any analyses at correct times.

5 FEATURES COMMON TO ALL APPLICATIONS

5.1 *Server functionality*

The MFM Suite applications include functionality to allow external network clients to download the currently open MFM project (process and

MFM models) and to access updated MFM function states and sensor values. The MFM Suite also accepts commands from external clients for setting MFM function states and sensor value ranges (Figure 10). This allows e.g. a field operator to have a situation understanding similar to a control room operator and to prune away cause paths that have been determined irrelevant.

5.2 *Web export*

The MFM Suite applications will provide updated displays of models with dynamic data accessible through an ordinary web browser. When opening an MFM project, the applications will generate several files, depending on the number of models. An example of the files is shown in Figure 11.

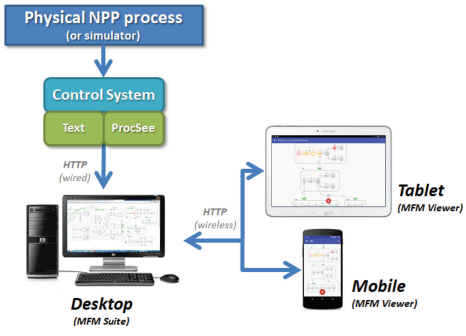

Figure 10. Data server connectivity.

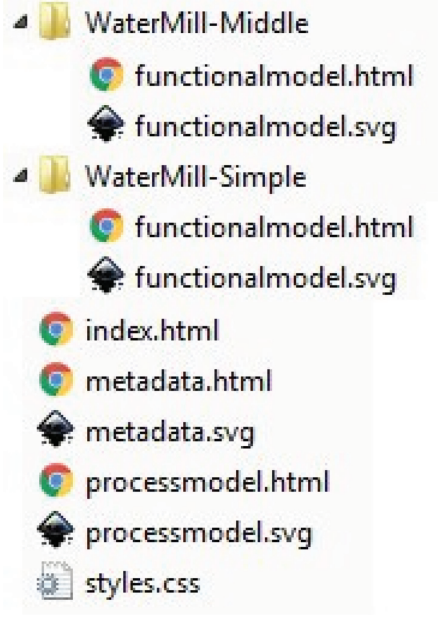

Figure 11. Web export files.

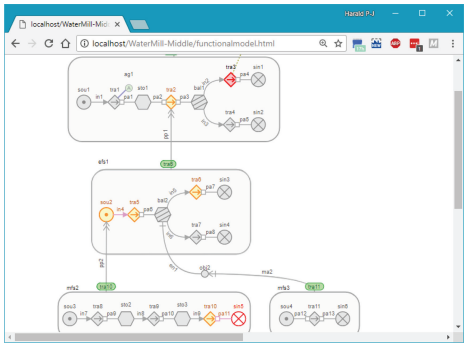

Figure 12. Web-page showing an MFM model.

The graphical models are exported as SVG (Scalable Vector Graphics) files, and since these graphics files are (as the name implies) scalable, they can be zoomed in the browser without the loss of any details (Figure 12).

Whenever the visual appearance of any element (MFM function, sensor etc.) in a model changes (e.g. due to an analysis affecting the elements), or whenever the user saves the project, the files are updated. Code is included in the HTML files to reload the pages at given frequencies (default every 5 seconds). This frequency is specified using the Settings dialog box of the applications, which also allows specifying an output folder, to which the generated files are exported. In order to make the web pages available to other devices in a network, this folder could be set to the active root folder of a web server application, such as e.g. Apache or IIS.

6 EXAMPLES OF USE

Several tests have been conducted using an MFM model of the primary side of two PWR simulators; the RIPS simulator which is based on the Ringhals power plant, and a generic PWR simulator (Zhang et al, 2014, Zhang et al, 2016, Thunem, 2017). Several scenarios were tested, including reactor trip cause by too low pressure in the steam generators, and a high pressure in the reactor coolant system.

Before running the experiment, the simulator was initialized to a normal running state. At this point, the values of all sensors were registered and used to set the individual alarm limits. The process model contained a total of 49 sensors, 21 of which were associated with MFM functions.

Before running each scenario, the simulator was reset to a normal running state. The control room operator would run the selected scenario by adjusting selected process parameters and letting the simulator run its course.

In most cases, the reasoning engine performed flawlessly. Figure 13 shows one scenario with multiple anomalous sensor readings (right side) and the successful diagnosis using MFM reasoning (left side).

It was, however, noticed that in some cases the selection of trigger function in the MFM model would cause a memory problem (stack overflow) in the reasoning engine. By manually selecting another trigger function, the problem would disappear.

The MFM reasoning engine uses the Java-based JESS inference engine (Friedmann-Hill, 2003), which employs recursion, a common cause of stack overflow problems. It is reasonable to assume that the selection of different trigger functions resulted in different recursion levels during the rule-based reasoning. One solution to this problem is to increase the Java call stack; however, the maximum size of the call stack may be platform dependent and difficult to ascertain.

Since the JESS system is no longer supported, the MFM reasoning engine has been re-implemented using the Drools inference engine, however, the experiments have not yet been re-run to verify any performance improvements.

7 MFM VIEWER

The *MFM Viewer* is a graphical model viewer designed for Android-based portable devices such as tablets and cell phones. The use of this application will facilitate a distributed team of e.g. control room and field operators to gain a common understanding of a process situation regardless of their location. For practical purposes a cell phone was used to create the images below, while in a work situation a tablet may be more suitable. The application will scale appropriately to all common display resolutions.

The main menu will allow the user to specify relevant settings, including the IP address and port number of the data server, i.e. any of the MFM Suite applications currently running. The main menu further provides a "Download project" menu item, which will instruct the MFM Suite applications to package the currently open project into a zip file and to transmit the zip file to the device. After receiving the zip file, the MFM Viewer will unzip the project files and display the project meta-model. Clicking on any of the model boxes (MFM or process) opens the model in a new window. In any of the models, the user may scroll and zoom using common finger gestures.

At the bottom of the MFM and process displays is a green button. Pressing this will cause the application to connect to the MFM Suite and retrieve and display the current MFM states and sensor values at a sampling frequency specified in the "Server Settings". The button will turn red, indicating the retrieval of data (Figure 14). Pressing the red button or returning to the meta-model will disable the connection. Note that for both models there is a search icon in the upper right corner. Clicking this will open a search field, which allows the user to find (by highlighting) any MFM function or process component whose outer label contains the given text string.

The MFM Viewer also allows the user to set the states of individual MFM functions or sensors. This is done by clicking on the desired element, which will open a dialog box, and clicking on the desired state (Figure 15). This will in turn trigger a re-analysis of the models in the MFM Suite application.

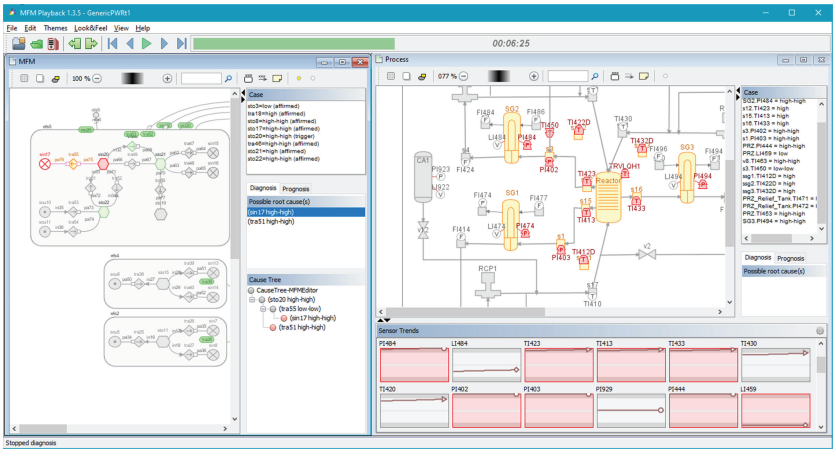

Figure 13. Scenario diagnostic.

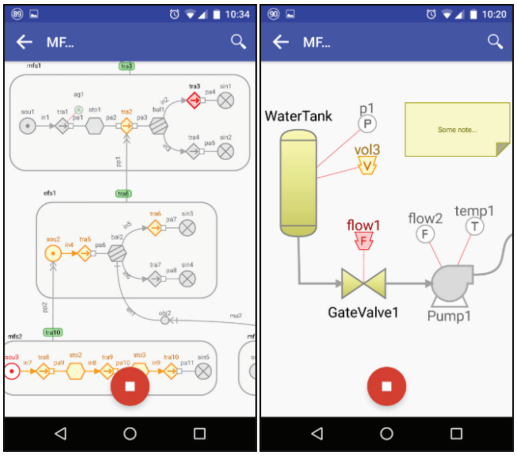

Figure 14. Displaying models and highlighting analysis results in the MFM Viewer.

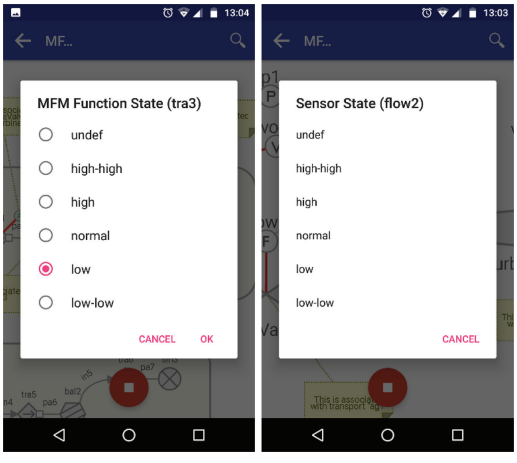

Figure 15. Setting MFM function and sensor states in the MFM Viewer.

8 FURTHER WORK

The primary focus on the MFM Suite development has until recently been on including necessary functionality to graphically design MFM and process models, create links between them, and perform online analysis with simulated process sensor data. When presenting the analysis results, focus has primarily been on how the MFM functions are affected, and which MFM functions have been identified as root causes. Interpreting the results has required some familiarity with the MFM methodology, which is unreasonable to expect from e.g. a control room operator.

To make the analysis results more accessible to someone without MFM training, the focus of the analysis presentation will therefore shift towards the process display, indicating which process components are the possible root causes and which components may be affected by the identified anomalies.

The activity will further evaluate various display modes to enhance the operators' comprehension of a given process situation in light of established analysis results, and to facilitate collaboration between control room and field operators via dedicated interaction mechanisms.

REFERENCES

Friedmann-Hill, Ernest, 2003. Jess in Action, *Manning Publishing Co* (ISBN 1930110898), USA, 2003 (http://www.manning.com/friedman-hill/).

Lind, Morten 2011. An Introduction to Multilevel Flow Modeling, *International Journal of Nuclear Safety and Simulation*, 2(1), pp. 22–32.

Thunem, Harald P-J et al. 2011. Using an Agent-oriented Framework for Supervision, Diagnosis and Prognosis Applications in Advanced Automation Environments, *Proceedings of the ESREL 2011 conference* (pp. 2368–2375, ISBN 978-0-415-68379-1), September 18–22, 2011, Troyes, France.

Thunem, Harald P-J 2012. Use-Case of an Agent-Oriented Framework for Supervision, Diagnosis and Prognosis Applications, *Proceedings of the combined ESREL2012/PSAM11 conference* (pp. 4910–4917, ISBN 978-1-62276-436-5), June 25–29, 2012, Helsinki, Finland.

Thunem, Harald P-J 2013. The development of the MFM Editor and its applicability for supervision, diagnosis and prognosis, *Proceedings of the ESREL2013 conference* (pp. 1807–1814, ISBN 978-1-138-00123-7), Sept 29 – Oct 2, 2013, Amsterdam, Holland.

Thunem, Harald P-J, 2017. Diagnostic Decision Support – Recent Development and Updates of the MFM Suite, *Halden Report HWR-1223*, Institute for Energy Technology, August 2017.

Thunem, Harald P-J & Zhang, Xinxin 2015. The continued development of the MFM Suite and its practical application on a PWR system, *Proceedings of the ESREL2015 conference* (pp. 2463–2471, ISBN 978-1-138-02879-1), Sept 7–10, 2015, Zürich, Switzerland.

Zhang, Xinxin 2015. Assessing Operational Situations, *PhD thesis*, Department of Electrical Engineering, Technical University of Denmark, June 2015.

Zhang, Xinxin et al. 2016. Applying MFM to the PWR Analysis: The experimental results and preliminary conclusions, *Halden Report HWR-1191*, Institute for Energy Technology, March 2016.

Zhang, Xinxin et al. 2014. Practical Application of the MFM Suite on a PWR System: Modelling and Reasoning on Causes and Consequences of Process Anomalies, *Halden Report HWR-1118*, Institute for Energy Technology, August 2014.