Proof of Concept of OpenMI for Visual DSS Development

[1]Dirksen P.W.,-[1]Blind M.W.,[2]Bomhof T.,[2]Srikrisnhudu Nagandla

[1]Institute for Inland Water Management and Waste Water Treatment RIZA, [2]University of Dortmund / Institute of Environmental Research

E-Mail**: p.w.dirksen@riza.minvenw.nl**

Keywords: *HarmonIT; OpenMI; DSS development.*

EXTENDED ABSTRACT

OpenMI (the Open Modelling Interface and Environment) is an application interface developed in the European project HarmonIT; it aims to standardize and simplify linking of (environmental/hydrological) models, databases, monitoring devices and thus allow (run-time) data-exchange, without additional programming effort. The main product of HarmonIT is a blue print of the standard interfaces. Software implementations, both in Microsoft .Net and Java, have been developed to proof its applicability. The establishment of the OpenMI will support and assist the strategic planning and integrated catchment management required by the European Water Framework Directive. However, OpenMI is not only applicable for linking hydrological models, but can be applied for development of visual decision support systems as well.

This paper will demonstrate a use case in which the OpenMI provides the communication between the model components Mozart and Agricom, and the AM-DSS-component (Agricom Mozart Decision Support System). The latter is the graphical user interface of the Agricom Mozart Decision Support System, whereas the former remain invisible to the non-specialist user. Mozart is a groundwater model for the unsaturated zone, which calculates water availability in different meteorological situations. Agricom is used to assess the agricultural and economic consequences of water management scenarios. The AM-DSS is developed as an OpenMI compliant linkable component that takes care of initializing and starting both Mozart and Agricom. The communication between the AM-DSS, Mozart and Agricom is based on the OpenMI application interfaces. The AM-DSS can be executed as a single application but can also be used as a component in a larger linked modelling system. Because AM-DSS is OpenMI compliant, OpenMI compliant tools such as the DataMonitor and EventViewer, can be used in a linked runtime environment.

The AM-DSS enables the user to define a number of scenarios in water management based on changed input to Mozart and Agricom. In addition, in Agricom the prices of different crops can be altered. In Mozart it is possible to change the land use, raise the riverbed level, change the drainage resistance or enable/disable sprinkling. Further measures can be included. For each scenario information on the costs of the measures are available.

AM-DSS runs the scenarios and compares the total yield and the prevented economic losses (i.e. economic benefits) of each scenario in relation to the current situation (change nothing) taking into consideration the costs of the measures. By that, the cost-benefit ratios for each of the scenarios can be estimated and an economically optimal water management strategy, crop selection and irrigation strategies at watershed level is possible.

One of the key benefits of OpenMI compliant models is that alternative models may easily be incorporated in model chains ("swapping models"). The AM-DSS however is a GUI/control and not a model, and selected options may affect both underlying model components. Though a 'generic' GUI is thus not feasible, the design and implementation focuses on minimizing the effort required if either underlying models are swapped for others. Likewise the AM-DSS can be used as part of a larger composition.

The driving forces for the development of AM-DSS are (1) to prove that OpenMI is not limited to linking models, but is also indispensable for DSS development and control and (2) the perceived need of agricultural DSS-systems to gain flexibility to avoid extinction.

This paper will provide details of designs and prove the usefulness of the OpenMI standard for DSS-development.

1. INTRODUCTION

The Water Framework Directive calls for integrated water management to be put into practice and identifies whole catchment modelling as a key part of integrated management. The only realistic mechanism for whole catchment modelling is integrated modelling in which models of different processes are linked together and hence allow process interactions to be simulated. OpenMI (the Open Modelling Interface and Environment; http://www.openmi.org/) is an interface being developed in the European project HarmonIT to standardize and simplify linking of (environmental/hydrological) models, databases, monitoring devices enabling OpenMI compliant models to exchange data as they run. HarmonIT (http://www.harmonit.org/) is a research project funded by the European Commission aiming at the development and implementation of a European Open Modelling Interface and Environment (OpenMI) that will simplify the linking of hydrology related models (Tindall *et al* 2005).

The OpenMI environment comprises a set of software tools (1: Sinding *et al* 2005, 2: Gijsbers *et al* 2005)). They facilitate making new and existing models OpenMI compliant. A graphical user interface and further tools facilitate linking and running models. A model system is OpenMI compliant when it implements the OpenMI Interfaces (Gijsbers 2005). The software delivered by the HarmonIT project is merely a proof of concept and one of the possible implementations of the OpenMI interfaces.

Integrated catchment management requires knowledge of all the processes and how these processes interact. This knowledge is too comprehensive to be possessed by any normal individual. Managers can only take a well founded decision when they are supported by a Decision Support System (DSS) managing the individual models of an integrated modelling system.

To date, DSS have tended to be used in addressing single issue problems such as flooding or drought, and several hard-wired, integrated systems have been developed. Integrated catchment management now requires a DSS to be able to represent multiple processes and how these processes interact. In other words, managers need a DSS with a large variety of models addressing multiple processes linked together and exchanging data on a timestep basis. OpenMI is designed to be used to link models in such a way, but is it also suited for developing DSS systems? Part of the HarmonIT project was to develop a very simple DSS based on the OpenMI software and Interface definitions. The Agricom Mozart DSS (AM-DSS) is the result of this effort and merely proofs the OpenMI concept rather than being intended as a real DSS. Mozart is a groundwater model for the unsaturated zone, which calculates water availability in different meteorological situations. Agricom is used to assess the agricultural and macro economic consequences of water management changes. Both Mozart and Agricom are OpenMI compliant model systems.

2. OPENMI

The aim of the OpenMI is to provide a mechanism by which physical and socioeconomic models can be linked to each other, to other data sources and to a variety of tools at runtime, hence enabling process interactions to be better modelled. Specific objectives are that the mechanism's design should:

- Be applicable to new and existing models
- Impose as few restrictions as possible on the modeller's freedom
- Be applicable to most, if not all, simulation techniques
- Require the minimum of change to the program code of existing applications
- Keep the cost, skill and time required to migrate an existing model to a minimum so that these factors are not a deterrent to the OpenMI's use
- Be easy to use
- Not unreasonably degrade performance

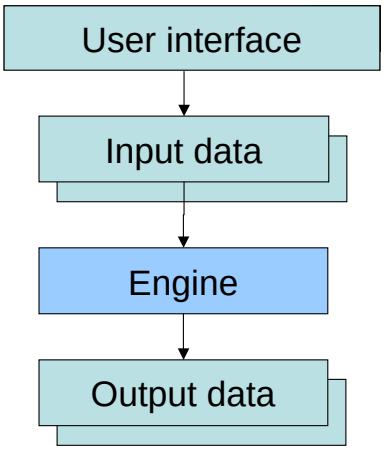

Figure 1. Modules of a model system

To date, almost all OpenMI compliant components are developed by wrapping legacy code (Fortran or C) using the software delivered by the HarmonIT project.

Most model systems have an architecture as shown in Figure 1. After wrapping, such a model system looks like Figure 2.

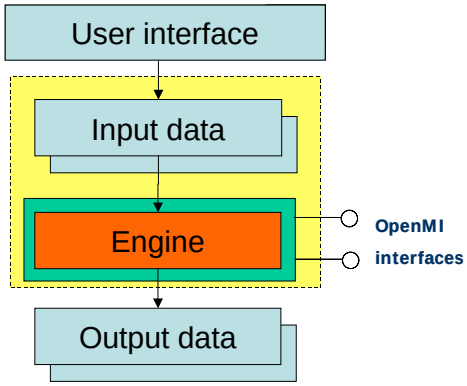

Figure 2. Wrapped model system

The model engine is re-structured and the OpenMI application interfaces are added.

Through the OpenMI application interfaces it is possible to (1) exchange data (input and output) and (2) control the component (initialize, compute timestep, finalize,etc). Particularly the second feature is essential when developing a DSS.

An OpenMI compliant component "tells" the environment which quantities at which locations can be delivered or accepted through the interfaces. The way to communicate this is by defining *Input-* and *OutputExchangeItems;* an exchange-item is a combination of a quantity and location, where the location can be one or more points, lines, polygons, polylines or polyhedrons. Linkable components can exchange data by a pull mechanism, meaning that a (target) component that requires input asks a source component for a (set of) value(s) for a given *ExchangeItem* for a given time. If required, the source component calculates these values and returns them. This pull mechanism has been encapsulated in one single method, the *GetValues()* method. Dependent on the status of the source component, this call may require associated computation and even more requests for data. An important feature is the obligation that components always deal with requests in order of receipt.

3. AM-DSS DESIGN

The main idea is to develop a controlling application (AM-DSS) that is capable of running Agricom and Mozart several times with adapted input based on scenarios selected by the user.

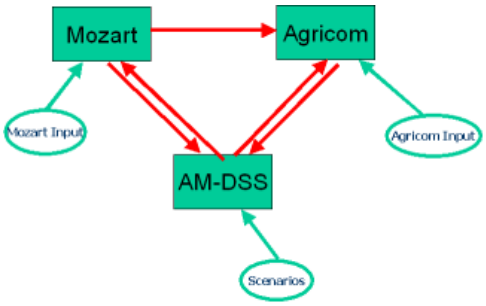

Figure 3. Setup of AM-DSS

Figure 3 shows the setup of the three components including the necessary input files.

The AM-DSS needs a graphical user interface enabling the user to select different scenarios and application dependent input. The AM-DSS graphical user interface is presented in Figure 4. As decision makers are not interested in models or model systems, the graphical user interface hides the components Mozart and Agricom, and offers only scenario selection and input of predefined quantities. The user interface has tabular input screens for every individual component; this facilitates possible future use of different model systems with similar functionalities.

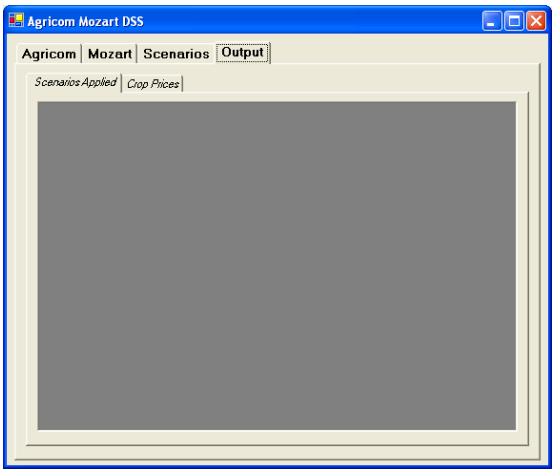

Figure 4. AM-DSS graphical user interface

The AM-DSS graphical user interface contains four tabular-sheets; one for Agricom related input (Figure 5), another one for Mozart related input (Figure 6), a scenario selection sheet (Figure 7) and a sheet with the output of the computations.

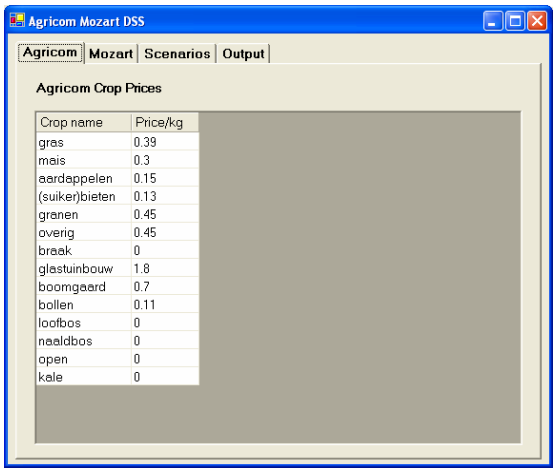

Figure 5. Agricom related input

Agricom uses different prices for different crops; in this module these prices can be overruled.

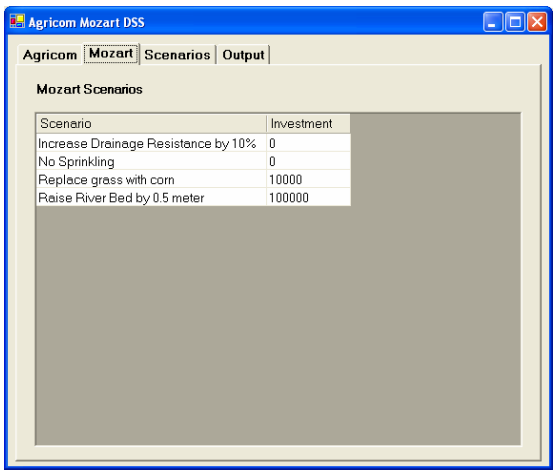

Figure 6. Mozart related input

AM-DSS contains 4 scenarios for adapting the input of Mozart. The Mozart input-files are very complex, so changing this input manually is error prone and needs extensive knowledge of Mozart.

For each scenario, experts have determined the necessary adjustments to the input files, which will be used to make the appropriate simulations. Economic experts have assessed the costs for implementation of these scenarios. Information about the adapted input files and the costs are read from an input file and AM-DSS implements the scenarios by editing the appropriate input files.

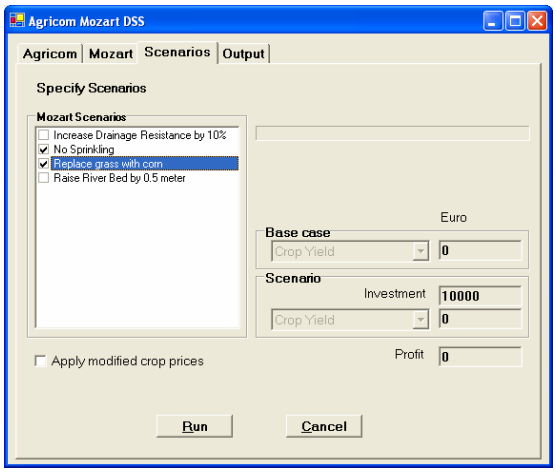

Figure 7. Scenario selection

The OpenMI application interfaces are used for running the scenarios and getting the results from the model systems. Figure 8 illustrates this first setup of AM-DSS.

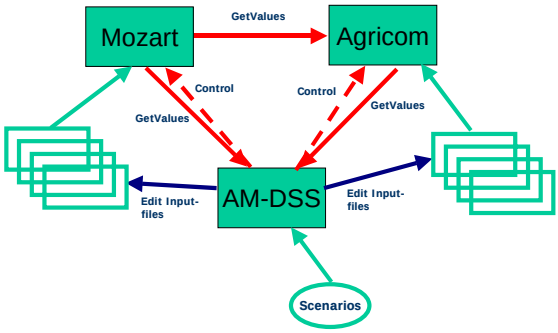

Figure 8. First implementation of AM-DSS

The scenarios module is the starting point of the model simulations. The user can select one or more Mozart scenarios and he can select to use the default crop prices or the prices adapted in the Agricom module. When selecting a scenario, the investment is altered based on the costs per scenario given by the expert. When the user presses the “Run” button, two computations are carried out; a base case and a case with the scenarios applied including the possibly adapted crop prices. As a result the net economic benefit is calculated based on the investment and the output of the two simulations.

AM-DSS runs the scenarios and compares the total yield and the prevented economic losses (i.e. economic benefits) of each scenario in relation to the base scenario (change nothing) taking into consideration the costs of the measures. By that, the cost-benefit ratios for each of the scenarios can be estimated and an economically optimal water

management strategy, crop selection and irrigation strategies at watershed level is possible. Results of different scenario runs are summarized in the output module. The user will get a good overview of the net economic benefit when applying different combinations of scenarios.

4. EXPERIENCES WITH AM-DSS

With this first OpenMI compliant DSS it is possible to make some pre-defined scenario computations and present the results to the user. At the time of developing the AM-DSS the components Agricom and Mozart were OpenMI compliant, but supported only a limited number of *ExchangeItems.*

The exercise proved that it is well possible to develop a DSS using the OpenMI interfaces and software. OpenMI has proven to be flexible and adequate enough to use in DSS development. The AM-DSS is designed to be a stand-alone application which instantiates the components Mozart and Agricom, generates the necessary links and controls running Mozart and Agricom with different sets of input. AM-DSS controls the model systems, edits the input given the chosen scenario and gathers information from the model systems for assessment of scenario impact. The AM-DSS is, on itself, a linkable component, enabling linkage of AM-DSS to other OpenMI linkable components.

This way of using OpenMI is rather different from the other use-cases of the HarmonIT project in which models were wrapped to be OpenMI compliant and computations were made using these components linked manually to other components through the OpenMI graphical user interface and running computations through this same user interface. In the case of AM-DSS, OpenMI software was used to instantiate the components, automatically generate the necessary links and run the computation. This approach underlines the fact that OpenMI is not a framework, but enables components to directly communicate with each other by means of the OpenMI interfaces.

The OpenMI components Agricom and Mozart were wrapped during the HarmonIT project even partly parallel to the development of the AM-DSS. At this moment Mozart and Agricom have a limited number of *ExchangeItems* and this prevents the development of a sophisticated DSS.

Mozart and Agricom are used by RIZA to make policy analysis for the Netherlands. The schematization of Mozart covers all hydrological units of the Netherlands. In case of the AM-DSS the schematization consists of only two hydrological units.

5. POSSIBLE ENHANCEMENTS

The input changes to the models Mozart and Agricom based on the scenarios chosen by the user are now implemented as changes to the input files of these components. This is not the way an OpenMI compliant DSS should work, but this method is used because Mozart and Agricom did not yet support enough *Input-ExchangeItems* to implement the input-changes through the OpenMI *GetValues* calls. When sufficient *ExchangeItems* are available, the AM-DSS software would be able to change the input values of the models by generating an OpenMI- link between the AM-DSS component and the model components for every model input quantity that is to be modified.

Eventually a relational database management system containing all input data of all linkable components might replace the input files now used by these components. Implementing this database system as an OpenMI linkable component enables the model components to query their input through the OpenMI *GetValues* method.

The setup of AM-DSS with different modules for different model components allows a model component to implement it's own user interface. One of the possible enhancements of the AM-DSS application is by providing a handle from Agricom to implement a user interface provided by Agricom replacing the current input of crop prices.

Of course further enhancement of the DSS with different model components or even an economic optimisation tool is possible, but this goes beyond the scope of the HarmonIT project. Proving the applicability of OpenMI for DSS systems is the main purpose of developing the AM-DSS

6. CONCLUSIONS

OpenMI has proven to be flexible and adequate enough to use in DSS development. The use-case also shows that being OpenMI compliant is not enough to ensure full implementation of a component in a DSS; the availability of *Input-* and *Output ExchangeItems* determines the applicability of components.

AM-DSS de facto contains the controller function implemented in traditional DSS and modelling frameworks. Due to OpenMI in general and the *GetValues* concept in particular, much bookkeeping is kept by underlying model

components. As a result, the controller function of OpenMI based DSS will be much simpler, especially when all I/O is accessible through the interfaces.

Developing applications like AM-DSS is not a big effort. The effort needed is even further reduced when model applications provide all necessary input and output exchangitems or even provide a graphical user interface which can be incorporated in the application to be developed.

7. ACKNOWLEDGEMENTS

The development of the AM-DSS and migration of Mozart and Agricom to OpenMI compatible components is financed by the European HarmonIT project.

8. REFERENCES

Tindall I. (editor), Moore R., Gijsbers G., Fortune D., Gregersen J. and Blind M. (2005), The OpenMI Document Series Part A – Scope (version 1.0).

Tindall I. (editor), Gijsbers G., Gregersen J., Westen S. Dirksen F., Gavardinas C. and Blind M. (2005), The OpenMI Document Series Part B – Guidelines to the OpenMI (version 1.0).

Gijsbers G. (2005), The OpenMI Document Series Part C – the org.OpenMI.Standard interface specification (version 1.0).

Gijsbers G. and Westen S. (2005), The OpenMI Document Series Part D – org.OpenMI.Backbone technical documentation (version 1.0).

Sinding P, Gregersen J., Gijsbers P, Brinkman R. and Westen J. (2005), The OpenMI Document Series Part F – org.OpenMI.Utilities technical documentation (version 1.0).

Dirksen F and Terveer R. (2005), The OpenMI Document Series Part H – Designs for additional OpenMI Tools (version 1.0).