LIANA Model Integration System - architecture, user interface design and application in MOIRA DSS

D. Hofman

Advances in Geosciences, 4, 9–16, 2005
SRef-ID: 1680-7359/adgeo/2005-4-9
European Geosciences Union

Advances in Geosciences

LIANA Model Integration System – architecture, user interface design and application in MOIRA DSS

D. Hofman

Studsvik Nuclear AB, 611 82 Nyköping, Sweden

Received: 1 August 2004 – Revised: 1 November 2004 – Accepted: 15 November 2004 – Published: 9 August 2005

Abstract. The LIANA Model Integration System is the shell application supporting model integration and user interface functionality required for the rapid construction and run-time support of the environmental decision support systems (EDSS). Internally it is constructed as the framework of C++ classes and functions covering most common tasks performed by the EDSS (such as managing of and alternative strategies, running of the chain of the models, supporting visualisation of the data with tables and graphs, keeping ranges and default values for input parameters etc.). EDSS is constructed by integration of LIANA system with the models or other applications such as GIS or MAA software. The basic requirements to the model or other application to be integrated is minimal – it should be a Windows or DOS .exe file and receive input and provide output as text files. For the user the EDSS is represented as the number of data sets describing scenario or giving results of evaluation of scenario via modelling. Internally data sets correspond to the I/O files of the models. During the integration the parameters included in each the data sets as well as specifications necessary to present the data set in GUI and export or import it to/from text file are provided with MIL_LIANA language. Visual C++ version of LIANA has been developed in the frame of MOIRA project and is used as the basis for the MOIRA Software Framework – the shell and user interface component of the MOIRA Decision Support System. At present, the usage of LIANA for the creation of a new EDSS requires changes to be made in its C++ code. The possibility to use LIANA for the new EDSS construction without extending the source code is achieved by substituting MIL_LIANA with the object-oriented LIANA language.

1 Introduction

The modern environmental decision support systems (EDSS) encompass the knowledge from different disciplines (such as geography, ecology, chemistry, risk assessment). This knowledge is utilised in the predictive models, GIS-based models and cost-benefit or multi-attribute analysis procedures (Haagsma, 1995; Erhardt and Shershakov, 1996; Monte et al., 2000; several systems presented in the Kovar et al., 1996; and Schulte et al., 2002). The complexity of each particular module of EDSS is very high and procedures of the development and validation are time-consuming. As result, the modules of EDSS are developed by the team of experts in the different disciplines and often are based of the legacy tools. For the development of the EDSS this leads to requests:

– System should be built as the integrated set of the independently developed tools. It is necessary to have models in such a form that it would be possible to validate and further develop the modelling, GIS, MAA or CBA tools both with and without connections with EDSS.

– It is required that users of the system will be supported by the friendly interface covering complexity of the tools and allowing user to concentrate on the problem and data instead concentration on software aspects of the EDSS

A successful attempt to satisfy these requirements had been done for the RODOS system (Bentz et al., 2001) where "models" are independent applications written in FORTRAN. Models exchange the data with the RODOS kernel via shared memory. The data to be exchanged with the kernel are defined in COMMON BLOCK section of the FORTRAN code of each model and system "knows" about them by using data descriptions provided during model integration. This method of integration gives big freedom to the developers of the models. The drawbacks however is too strong connection of the integration methods with the features of the OS Unix and programming language as well as necessity to emulate the working of the kernel for the standalone running of the models.

LIANA Model Integration System (Hofman, 1998a; Hofman, 1999) was developed as the shell application for the EDSS with the following features:

Correspondence to: D. Hofman
(dmitry.hofman@studsvik.se)

– It can be quickly integrated with standalone models or other application such as GIS and databases. On the Windows system, a "model" can be any Windows or DOS.exe application retrieving input and producing output as text files.

– It contains a built-in data-centred (Microsoft, 1998) user interface allowing data browsing by means of a desktop metaphor and by selection from the map.

Basic ideas of LIANA had been tested on the Unix platform during development of the first version of the standalone Hydrological module of the RODOS system (Hofman et al., 1996). In the frame of MOIRA EC project (Monte et al., 2000) LIANA has been redeveloped for the Windows platform using Visual C++ and implemented as the MOIRA Software Framework (MOIRASF) (Hofman et al., 2000). MOIRASF is the "operating system" and user interface of the MOIRA Decision Support System.

The article discusses the architecture and functionality of the LIANA system, its application for MOIRA DSS and extensions to LIANA that need to be implemented to simplify its application for the construction of new EDSSs.

2 Architecture and functionality

The basic principles of LIANA functionality are shown in Fig. 1. This functionality is supported by the framework of C++ classes and functions covering most common tasks required for the EDSS (such as managing of scenarios and alternative strategies for each scenario; running of the chain of the models; keeping consistency between changes in scenario data and results; access to data in Explorer-like and GIS-like styles; visualisation of the data with tables and graphs; supporting of the data base of reference values and ranges for the parameters etc.) The simplified class diagram of LIANA is shown in Fig. 2.

2.1 Data objects

The key objects for the LIANA system are the objects of the class *Data*.

For the user the EDSS is presented as a hierarchy of *Data* objects. *Data* contains values for one or more parameters related to the description of scenario to be evaluated by EDSS or results of the evaluation of scenario with modelling. In addition to the values of the parameters *Data* contains the information about source of the values (for example "estimated by the GIS", "default value").

From the integration point of view each *Data* contains information necessary to prepare one of the input files of the modes or information imported from one of the output files of the models.

The lifetime of a *Data* object is normally limited to the current function or the lifetime of the user interface object visualising *Data*. The system stores and retrieves content of the Data to/from file using methods similar to ones implemented in Document/View architecture (Microsoft, 1995). The file used to keep *Data* in persistent form called in LIANA a "data set".

2.2 Data classes

Each data set used in the system has a corresponding "data class definition" which is described using the MIL_LIANA language and contains:

– Type and basic properties of the data set (such as for example "input", "output", "countermeasure", "editable" etc.)

– Types, names and initial values of variables of the data set

– Format of the table presenting the data set in the GUI

– Format specification for importing or exporting a data from/to I/O file of the model

After the first request to data set class definition it is parsed and stored in an internal format, therefore parsing is required only once for each class definition.

2.3 Scenarios and alternatives

The objects of classes *Solution*, *Alternative* and *Results* are the lists. They contain filenames and GUI-related run-time information for the data sets. The framework uses serialization mechanisms provided by MFC to load and store these objects at run-time. The configuration files are used to describe the content of each list.

The objects of class *Alternative* keep information about data sets defining alternative strategies (for example alternative strategies of countermeasures that can be applied for the lake contaminated by radionuclides). The content of *Alternative* is dynamic. User can select which data sets are to be included in each *Alternative*. System contains the configuration file describing all possible "countermeasure" data sets and their "compatibility" (some of countermeasures can not be applied simultaneously).

Each *Alternative* has corresponding *Results* containing filenames of data sets related to the results of the evaluation of *Alternative* via modelling. The objects of classes *Model* and *Chain* provide the functionality related to model chain execution and obtaining of results of the simulation.

The *Solution* keeps information about data sets valid for all *Alternatives* (such as site-specific data) as well as references to the *Alternatives*. All data sets related to *Solution* are stored in the separate directory. Data sets directly enumerated in the *Solution* are stored in the root of this directory. A new subdirectory is created for each *Alternative*. LIANA can manipulate *Solution* directories as unit objects (such as it allows to perform operations "Open", "Open last", "Save as..." etc.).

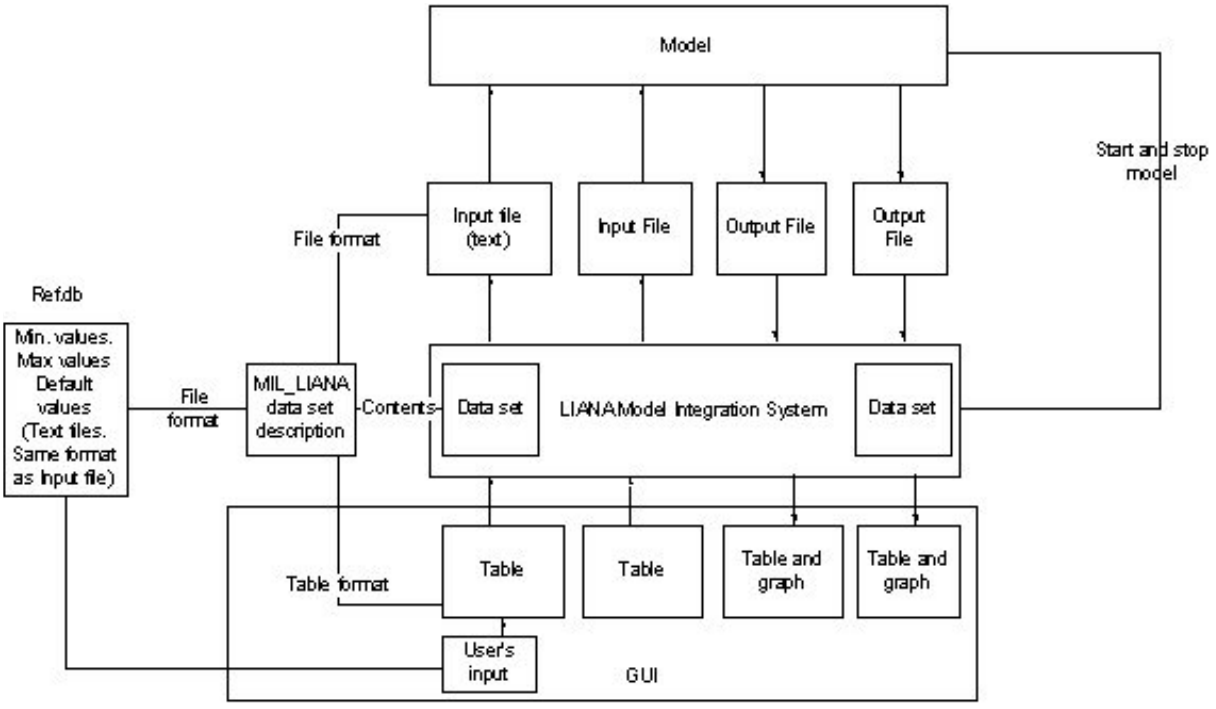

Fig. 1. Basic principles of LIANA functionality.

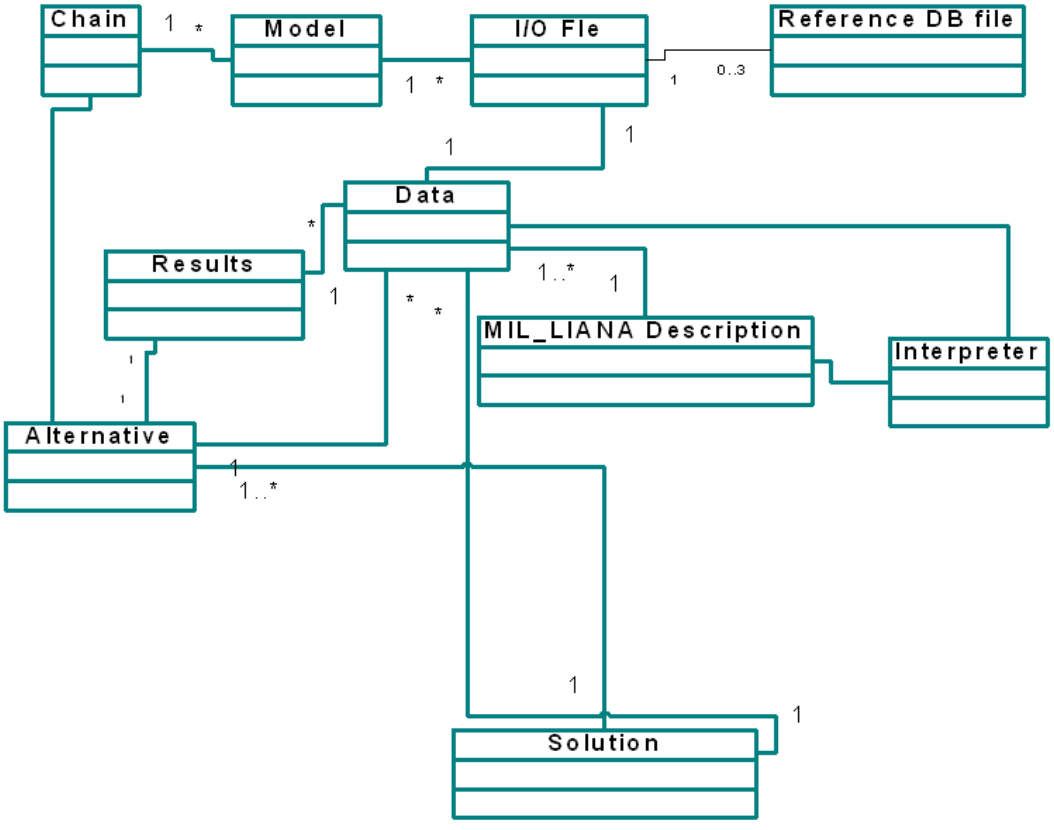

Fig. 2. Simplified class diagram of LIANA.

2.4 Model integration and EDSS construction

EDSS is constructed by integration of LIANA system with the models or other applications such as GIS or MAA software. The basic requirements to the model or other application to be integrated is minimal – it should be a Windows or DOS .exe file and receive input and provide output as text files. Each I/O file of the model corresponds to one of the data sets. The kernel of the system "knows" about the format of each I/O file of the model by parsing and using the corresponding data class definition. MIL_LIANA definition can describe complex formats of the file with "scalar" data, but it is assumed that a file with time-dependent data has a column-based structure.

The integration means creating of the MIL_LIANA files and configuration files describing content of *Solution, Alternative* and *Results*. During the integration it is also necessary to make a changes in LIANA source code in order to describe:

1. Manipulation with data sets other than import and export to/from the files and time interpolation of the data.
2. Import or export from and to files with very complex format
3. Data query from several sets
4. Relationship between data and models
5. Conditions for the execution of the certain model in the model chain

At present the work to describe the features of the EDSS enumerated above with the data class definitions and configuration files and without necessity to make a changes in the C++ code is in progress.

2.5 User interface

LIANA system assumes that content of all lists (such as *Solution, Inputs, Alternative, Results*) is available for the users (via GUI tools) in Explorer-like style as icons (or buttons) or in GIS-like style by the clicking on the map. "Activating" of the certain data set referred in the list (for example by clicking on the icon) may results in:

Data set type	**Data set does not exist**	**Data set exists**
"Input"	Opening the UI tool, showing template and initial values (or "not defined") for the data set and giving the possibility to edit it.	Opening the UI tool, showing of the data set content and giving the possibility to edit it.
"Output"	Starting the chain of models (via Alternative object) to obtain data set information.	Opening a corresponding UI tool, showing of the data set content without the possibility to edit it.

Several sets of "LIANA-compatible" user interface tools (on the DOS, Unix and Windows platforms) supporting functionality described above had been developed.

Developers of each model affect the appearance of the GUI of EDSS when decided how to split the model data in different files. For example the input or output data for the particular model can be presented in GUI as one long table (containing all of the parameters) or as number of the short tables each related to one of the parameters. Developers of the model will make the decision about it. Such participation helps to construct the user interface quickly and utilise the broad experience of the developers in a scientific subject and their experience with the communication with users of the particular model.

The developers of models also supply the information necessary to maintain the database of reference values (RefDB). During the integration each model can be optionally supplied with

- Files with the default values for certain parameters
- Files with the minimum values
- Files with the maximum values

These files should have the same format as the corresponding input file of the model (and thus available for the kernel with the same MIL_LIANA description). Default values for the parameters are available for the user while working with the GUI tables. Range values are utilised to check user's input.

3 Application

The MOIRA DSS (**Mo**del-Based Computerised System For Management Support To **I**dentify Optimal **R**emedial Strategies For Restoring Radionuclide Contaminated **A**quatic Ecosystems And Drainage Areas) helps decision-makers to evaluate and rank alternative strategies which could be implemented to reduce consequences of accidental radioactive contamination of lakes and rivers. The evaluation of each strategy (including "no actions") is done in MOIRA in terms

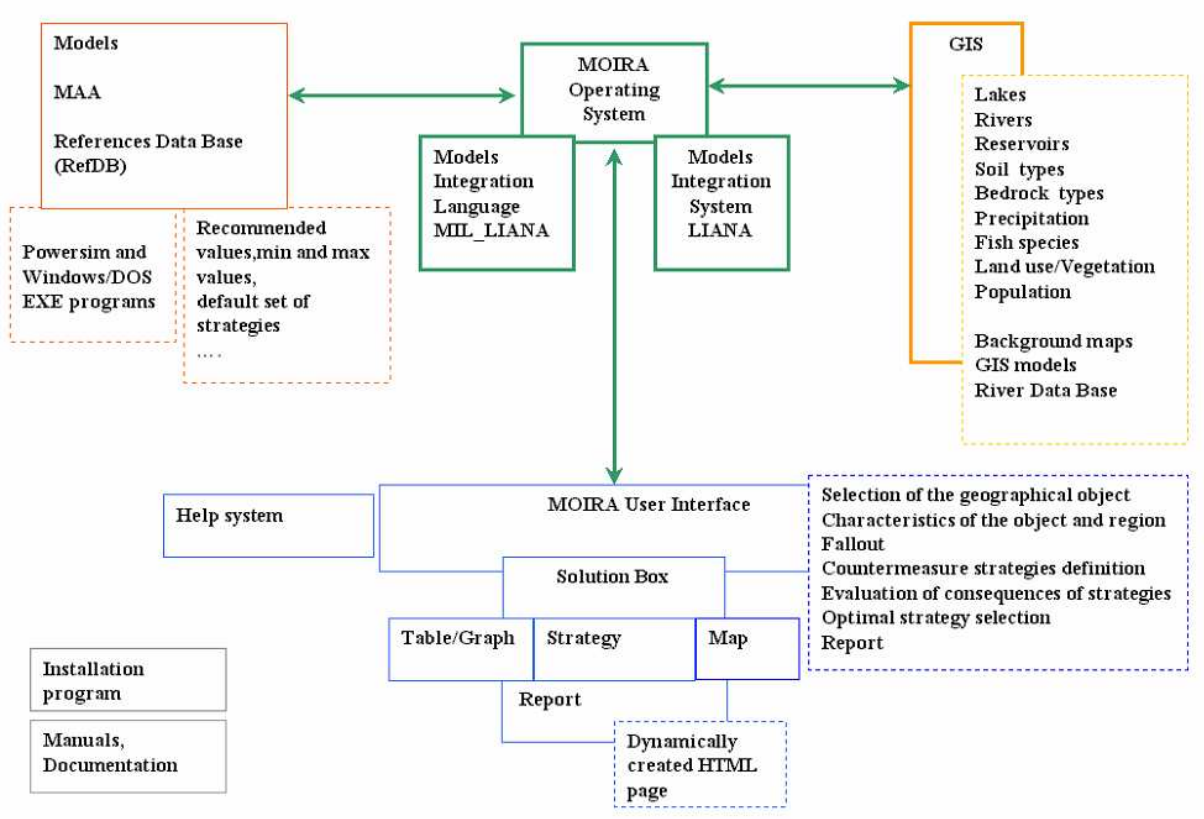

Fig. 3. Architecture of the MOIRA DSS.

of doses received by population as well as environmental and economic consequences. To achieve its goals MOIRA includes a set of models to predict the distribution of Cs-137 and Sr-90 in lakes and rivers, models evaluating chemical hydrological and characteristics of the lake water, lake ecosystem index model, dose assessment model, economic model. Ranking of the different strategies is made using Multi-Attribute Analysis (MAA) techniques. MOIRA models and MAA module are described in several sources (Monte and Brittain, 1998; Monte et al., 2000; Håkanson et al., 2000; Gallego et al., 2002). The models was implemented in the environment of Powersim® – or as Windows or DOS applications.

During the MOIRA project collection of environmental and population data was carried out using MapInfo GIS. GIS-based models were developed (using MapBasic®) to help the user in querying or estimating of GIS-based data related to a certain lake, part of a river or another geographical location.

The MOIRA system is the result of the design, development and evaluation undertaken during the MOIRA (Monte et al., 2000) project, COMETES (Monte et al., 2002a) project, TRA-RAD-FW course (EC-Sponsored training course on radiological assessment and decision-making for the management of contaminated freshwater ecosystems, Madrid, Spain, 2002) as well as ongoing EVANET-HYDRA thematic network http://info.casaccia.enea.it/evanet-hydra. MOIRA software and documentation web-site is http://moiradss.topcities.com.

Due to the necessity to implement in MOIRA of several models reflecting different fields of the knowledge (radioecology, chemistry, dose assessment, economic, limnology, multi-attribute analysis) and development of MOIRA by an internationally distributed team of experts, it was decided that the component architecture and highly flexible procedure of integration of modelling components is most suitable for the development of the system (Appelgren et al., 1996; Hofman, 1998). As result the research version of LIANA Model Integration System was used in order to create the MOIRA Software Framework (Hofman, 1998). Further updates of the MOIRA Software Framework (MOIRASF) are described in (Gallego et al., 2002b) and in on-line documents (Hofman, 2003, 2004).

The architecture of the MOIRA DSS is shown in Fig. 3 from Hofman et al. (2000).

In addition to the “usual” .exe applications MOIRASF version of LIANA has the possibility to integrate MapBasic® – models (using connection with MapInfo server based on OLE Automation (MapInfo, 1995)) and contains a (.exe) “proxy” to integrate Powersim-based models. These types of models should also receive input and produce output as the text files.

Data-centered design of MOIRA GUI (Fig. 4) provided by LIANA satisfies the requirement (Appelgren et al., 1996) that the MOIRA should be a friendly system for the users with the different level of experience in environmental modelling.

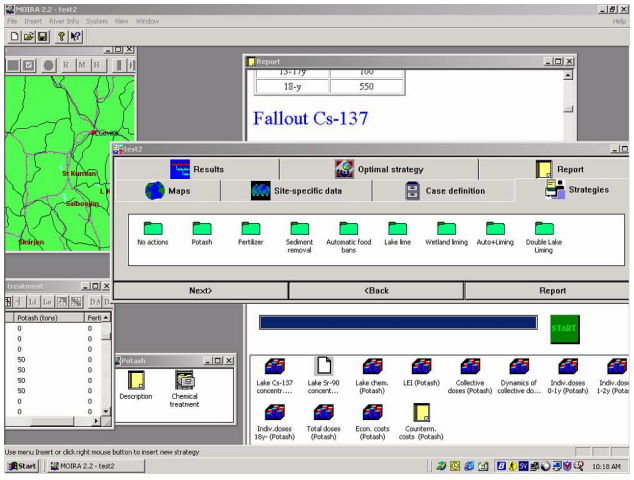

Fig. 4. MOIRA user interface.

With the GUI the user can start from the direct request of the data of interest (it could be for example "ranking of the alternative strategies" for the decision maker or "concentration of Sr-90 in the water" – for expert in radioecology). Windows (Visual C++) implementation of LIANA in MOIRASF contains the GUI classes inherited from MFC CView class. For example class MapView (supported by MapInfo Integrated Mapping) gives the user the option to browse data sets linked to the geographical location by clicking on the map; class TextView gives the possibilitry to preview the text files and supply each Alternative with the short text comment

The following advantages for the development process using LIANA system was notified by the author:

– Limited requirements of the LIANA to the model allowed to integrate the models in the form they are received from developers.

– After the integration the same versions of the models and MAA software was used both in the frame of MOIRA and stand-alone. This allowed developers of the models to test and further develop the models using known software environment without necessity to learn new things related to the MOIRA architecture and functionality. This gives the possibility for the simultaneous and independent development of the models and "kernel" of the DSS.

– The updated versions of the models was integrated just by substituting of the corresponding .exe, .sim or .mbx file. If new data was used or produced by the updated version of the model then in most cases only the changes in data class definition files and configuration files had been required. The GUI appearance had been changed automatically in response to the changes in these files.

4 Discussion

The future applications of LIANA (in its MOIRASF version) for the creation of a new EDSS are possible via either:

– Using its Visual C++ code as the general-purpose object-oriented C++ framework for creating the EDSS system kernel and user interface.

Or

– Using LIANA in its compiled form as the "shell" application allowing the construction of EDSS without making changes in the framework's source code – only by means of the description and configuration files.

At the moment even in the case of the second option some changes in the C++ code may be necessary. The integration of a new model with LIANA only by means of data class definition and configuration files is achieved by substitution of the MIL_LIANA descriptions to ones written on the LIANA language (LL) (Hofman, 1999). The LL is the full-scale build-in object-oriented programming language developed for the LIANA. The MIL_LIANA can be considered as its simplified subset. Data class definition in LL is a C++ like definition, which however may contain the following additional sections:

– *Produces* – class members available for access from other *Data* objects. Only these values will be saved to the data set.

– *Needs* – the other objects (data sets), which must be prepared before execution of the command described in *Realization*. Conditional creating of the certain data objects can be specified by using "if" statement.

It is easy to see that creating an object described by the LL class (with the non-empty *Needs* section) will automatically construct the necessary chain of models.

– *Realization* – for "output" data sets this section includes statements that start external model. One of the output files of this model will contain the data defined in section Produces which will be imported to the data set using information from Represented section. Conditional run of the model or run of alternative models can be specified with "if" statement. Alternatively, Realisation can include only the statements for pre-processing data from other data sets (specified in Needs) followed by SAVE command. For "input" data sets Realization is empty or contains statement which issues a message to the user.

– *Represented* – LL statements or a MIL_LIANA specification for exporting or importing data set information (given in *Produces* section) to or from I/O file of the model.

The GLOBAL and LOCAL keywords used in the LL data class definition help to specify the location of the data set (directory related to *Solution* or *Alternative*) and name of the data sets.

– *Table* – MIL_LIANA format of the table presenting a given data set in the user interface.

In addition to its advantages related to the rapid integration of models with LIANA the LL programs can be used as an alternative to the user interaction with GUI which will give the possibility to implement in EDSS an automatic mode of execution similar to the one used in the RODOS system (Steidlinger and Bentz, 2003).

Even after incorporation of LL the integration of the models presented in the form other than .exe files, PowerSim files, MapBasic files or performing data exchange with the methods other than text files will require the changes to be done in LIANA source code or writing of the wrapper (.exe) application around the corresponding model. This problem can be solved by incorporating to LIANA the interfaces to the most common integration standards emerging or to be emerging in the modern EDSS (Argent, 2004; Blind and Gregersen, 2004). The interfaces can be developed either by extending of the C++ framework with new classes (in similarity with introducing of the MapBasic® – models integration in MOIRASF) or "proxy" .exe applications integrated with the system (in similarity with the Powersim® – models integration in MOIRASF).

5 Conclusions

LIANA could be used as the basis for the creation of a wide range of model-based systems on the Windows platform.

The architecture, model integration techniques and user interface design implemented in LIANA helped to quickly create the reliable user-friendly decision support system – MOIRA DSS.

Despite its high flexibility, currently using the LIANA for the creation of new EDSS requires making changes in its C++ code. The future extensions of the system will be concentrated on the providing of the possibility to construct new EDSS by integration of models with LIANA only by means of the data description and configuration files.

6 Software

The LIANA (in its MOIRASF version) is available as the part of the MOIRA system at http://moiradss.topcities.com.

Acknowledgements. The author thanks M. Zheleznyak (IMMSP, Ukraine) for the supervision in the work (Hofman, 1999) and to Y. Sorokin (IMMSP, Ukraine) and L. Papush (currently Linköping University, Sweden) for the valuable discussions related to the LIANA system design and features of the LIANA language. Architecture, model integration techniques and user interface design of the MOIRA Software Framework has been constantly assessed by A. Appelgren, U. Bergstrröm and S. Nordlinder (Studsvik, Sweden), J. Brittain (University Oslo, Norway), E. Gallego (UPM, Spain), R. Helling (NRG, the Netherlands), L. Håkanson (Uppsala University, Sweden) and L. Monte (ENEA, Italy). Evaluation of software features of the MOIRA Software Framework and planning of its future development was performed in the framework of EVANET-HYDRA EC project. Thanks to K.-E. Lindenschmidt (GeoForschungsZentrum Potsdam, Germany) for his editorial help with the manuscript.

Edited by: P. Krause, S. Kralisch, and W. Flügel
Reviewed by: anonymous referees

References

Appelgren, A., Bergström, U., Brittain, J., Gallego, E., Håkanson, L., Heing, R., and Monte, L.: An outline of a model-based expert system to identify optimal remedial strategies for restoring contaminated aquatic ecosystems: the project "MOIRA", ENEA Technical Report, RT/AMB/96, ISSN/1120-5556, Rome, 1996.

Argent, R. M.: An overview of model integration for environmental applications-components, frameworks and semantics, Environmental Modelling & Software, 19, 3, 219–234, 2004.

Bentz, G., Lorentz, A., Rafat, M., and Schüle, O.: Integration of external programs in RODOS, RODOS-TN(1)9601, 2001.

Blind, M. and Gregersen, J.: Towards an Open Modelling Interface (OpenMI), in: iEMSs 2004 International Congress: Complexity and Integrated Resources Management, edited by: Pahl-Wostl, C., Schmidt, S., and Jakeman, T., International Environmental Modelling and Software Societey, Osnabrueck, 2004.

Erhardt, J. and Shershakov, V. M.: Real-time on-line decision support systems (RODOS) for off-site emergency management following a nuclear accident, Final Report, EUR 16533, pp. 1–118, 1996.

Gallego, E., Jiménez, A., and Hofman, D.: The version 2 of MOIRA system, A direct result of COMETES project, in: COMETES Final Report (Ref. Monte et al.), pp. 179–184, 2002.

Gallego, E., Jiménez, A., Mateos, A., Sazykina, T., Rios-Insua, S., and Windengård, M.: Application of Multiattribute analysis (MAA) methodologies to the evaluation of the effectiveness of remedial strategies with the Moira system, in: COMETES Final Report (Ref. Monte et al.), pp. 11–120, 2002.

Haagsma, L.: The integration of computer models and data bases into a decision support system for water resources management, Modelling and Management of sustainable Basin-scale Water Resources Systems, Proccedings of a Boudler Simposium, IAHS Publication, No. 231, pp. 253–261, 1995.

Håkanson, L., Gallego, E., and RÍos-Insúa, S.: The application of the lake ecosystem index in multi-attribute decision analysis in radioecology, Journal of Environmental Radioactivity, 49, 3, 319–344, 2000.

Hofman, D., Lyashenko, G., Marinets, A., Mezhueva, I., Shepeleva ,T., Tkalich, P., and Zheleznyak, M.: Implementation of the aquatic radionuclide transport models RIVTOX and COASTOX into the RODOS system, in: The Radiological Consequences of the Chernobyl Accident, edited by: Karaoglou, A., Desmet, G., Kelly, N. G., and Menzel, H. G., Proceedings the Radiological Consequences of Chernobyl Accident, Proceedings of the first international conference, 18–22 March 1996, Minsk, European Comission, Luxenbourg, pp. 1181–1184, 1996.

Hofman, D.: Usage of software system LIANA for the integration of application tasks, GIS and data bases into the decision support systems based on models, Mathematical machines and systems N1, Kiev, pp. 75–88, 1998.

Hofman, D.: Design and development of the MOIRA software framework, in: Principles for the development of the MOIRA

computerised system (Ref. Monte, Brittain), pp. 37–57, 1998.

Hofman, D.: Software framework for integration of the models into the decision support system on ecological safety, PhD dissertation, Kiev, 1999.

Hofman, D., Appelgren, A., Bergström, U., Karlsson, S., Mathiasson, L., and Nordlinder, S.: The MOIRA software framework, geographical information system and integration of models, in: MOIRA Final Report (Ref. Monte), 2000.

Hofman, D.: New features of the MOIRA 2.2., http://moiradss.topcities.com, 2003.

Hofman, D.: List of the changes http://moiradss.topcities.com, 2004.

Kovar, K. and Nachtnebel, H. P. (Eds.): Application of Geographic Information Systems in Hydrology and Water Resources Management, Proccedings of HydroGIS 96, IAHS Publication No. 235, 1996.

MapInfo Corporation: MapBasic Users's Guide, Troy, New York, pp. 279–321, 1995.

Microsoft Corporation: Programming with MFC, Microsoft Press, 1995.

Microsoft Corporation: The Windows Interface Guidelines for Software Design, MSDN Library, 1998.

Monte, L. and Brittain, J. (Eds.): Principles for the development and implementation of the MOIRA computerised system. ENEA Technical Report RT/ABM/98/4, ISSN/1120-5555, Rome, 1998.

Monte, L., Van der Steen, J., Bergström, U., Gallego, E., Håkanson, L., and Brittain, J.: The project MOIRA a model-based computerised system for management support to identify optimal remedial strategies for restoring radionuclide contaminated aquatic ecosystems and drainage areas, Final Report, ENEA Report, RT/AMB/2000/13, 2000.

Monte, L. , Brittain, J., Håkansson, L., Gallego, E., Zheleznyak, M., Voitsehovich, O., Kryshev, I., Marinov, and Petrov, K.: Implementing Computerised Methodologies to Evaluate the Effectiveness of Countermeasures for Restoring Radionuclide Contaminated Fresh Water Ecosystems, COMETES Project, Final Report, Ente per le Nuove Tecnologie, l'Energia e l'Ambiente (ENEA), RT/AMB/2001/28, Roma, 2002.

Schulte, E.-H., Kelly, G. N., and Jackson, C. A. (Eds): Decision support for emergency management and environmental restoration, EURATOM, EUR 19793, Belgium, 2002.

Steidlinger, A. and Benz, G.: RODOS User Guide. System Interface, http://www.rodos.fzk.de/RodosHomePage/RodosHomePage/Documents/Public/HandbookV5/UserGuide/SystemUser_5.pdf, 2003.