Developing environmental model bases

G. Guariso, A. Rizzoli
Department of Electronics and Information, Politecnico di Milano, Milan, Italy

Abstract

Though a systematic method for the design of model bases is not yet available, the paper revises a number of recent experiences. They show that database techniques must be integrated with expert systems and hypermedia to support model selection and application by inexperienced users efficiently and correctly. In particular, it appears that a database approach is preferred when dealing with a very large and heterogeneous set of models, while rule base systems better support model selection in a restricted knowledge domain and hypermedia can be the best tool to supply experience on model use. In most cases also object-oriented techniques are very useful.

1 Introduction

In a recent survey among environmental analysts, the availability of computerised bases of environmental models was ranked as a first priority by 23% of the interviewed. Indeed, a routinely application of mathematical models and related software for the simulation and the management of natural resources is still prevented by the difficulties encountered in selecting and properly use the available tools.

The storing and retrieval of information related to a model is more complex than that traditionally used, for instance, to search for a book or to register population data. Models are active components that must in turn provide significant answers to users' problems and thus their management requires specific tools. The problem is even more relevant if the user wants to find models that describe different compartments of a complex environmental system and wants to link them together to explore alternative actions or decisions on that system. For instance, in a model of lake eutrophication, one

sub-model can describe limnological processes and another ecological processes, the latter can also be decomposed into sub-models and these models may in turn be used within a mathematical programming program to find optimal treatment decisions according to a given performance criterion.

When a modeller starts working on a new environmental problem different approaches may be selected:

- pre-packaged routine modelling;
- innovative modelling:
- multi-domain model integration.

The first case is the most common: the problem is well structured and widely studied. Other modellers in the past have produced standard methodolgies and implementations to solve the modelling problem. The problem is, first, to find out the model(s) suitable for that problem and then to convert the data in the format required by the specific model implementation. These problems are solved, at various levels of user friendliness, by *model directories* and *model libraries*, which are characterised by the ability of storing and retrieving a variety of programs each with specific characteristics and a fairly rigid structure.

The second case, despite being the hardest from the modelling point of view, is the most welcome by model definition and integration theorists. They are in fact free to ask the modeller to write the model using their preferred tools and languages that allow for perfect integration and re-use of the newly created pieces of work. Only rarely, however, the amount of really innovative work written in the model would justify its complete rewriting. All models are based on many years of previous work and therefore they also must re-use older components. It is only a limit of present model development packages the fact that it is often easier to rewrite everything rather that to adapt existing models.

In the third case, the modeller has already browsed through model libraries and found all the relevant ingredients for the job. The problem is thus to assemble these models, sometimes written in different programming languages or operating systems, imposing different requirements on data format, and to make them work together.

As noted by Dolk [1], research in the last ten or fifteen years has mainly concentrated on the development of model management systems (MMS) to better support the creation of new models and their use. In fact, a summary of the desirable features of MMS, recently given by Potter et al. [2] on the basis of several previous works, lists the following points:

1) The MMS should be able to create new models quickly and easily.
2) The model building-blocks of MMSs should be cognitively meaningful chunks of knowledge to the user.
3) MMSs should be able to inter-relate models with appropriate linkages, thus providing the functions of model integration and model decomposition.
4) MMSs should be able to manage the model base with functions analogous to data base management.

5) MMSs must have a meta-level encyclopaedia, analogous to a DBMS's data dictionary, which includes a repository of data, heuristics, tasks, models, and the relationships between them.

Some packages implementing these ideas have already become commercially available (e.g., STELLA™ or Simulink™). However, to support the third kind of modelling listed above, MMSs should have another important feature: the ability to incorporate executable models written by other modellers and to connect them in a seamless way. This is essential in the environmental domain that has a long tradition of model development and some programs have been applied in hundreds of studies, thus becoming a legacy of primary importance. Research in this area is only at the beginning and aims at making model re-use more convenient than model re-write.

2 Pre-packaged models

The literature on environmental models has grown in recent years to considerable dimensions. Recent surveys about available software implementing such models quote numbers of the order of several hundreds for groundwater problems [3], more than hundred for air pollution [4,5], few tenths in areas like ecology or surface water management [6]. The problem of finding the one(s) suitable for a given application is thus becoming very difficult.

2.1 Management of models with a fixed structure

Lists of available models have been compiled, mainly in paper form, since long ago. The Environmental Software Directory [7] and Elsevier's Report on Environmental Software [8] are examples of this approach. Only since very recently, however, these libraries are being supplied in a digital form on high volume storage devices such as CD-ROM's [6]. These applications have been mainly developed for modelling experts and thus provide only moderate support to the model screening phase, when the user is looking around for a model that fits his/her needs.

Designing a database for model selection has a number of different and sometimes conflicting objectives. The information should be highly structured so that the computer could easily access it and should be highly flexible to allow easy interaction with the non computer-oriented user. The model must be well described so that the user can get at least some idea of its suitability for his/her specific application, but should be short enough to allow a fast browsing among a number of similar models.

The LIVIA modelbase [9] was built for a wide target audience with quite different, and usually rather basic computer backgrounds. It was implemented on a personal computer together with facilities for model validation and comparison. At present, the model base consists of about 100 well tested and documented programs for which the executable code is also available. They

were acquired from various private and public environmental agencies and cover the following main environmental fields: ground water (17 models), air quality (31), dynamics of natural ecosystems (12), hydrology (10).

The information about models in the LIVIA base was structured into four different levels. The first level corresponds to a short description of the model characteristics by the use of a number of *keywords*. These represent a strongly condensed form of description, but they allow a fast search operation and the possibility of displaying the basic information about a model on a single computer screen. For instance, to search a model via keywords, the user can specify one or more values referring to one or more of the following aspects: model name, model objectives, application field, techniques used, type of computer. This structure allows all the normal database operations such as the insertion of new models/values as well as the retrieve and deletion of existing ones.

At the second level, one can find the information organised in a sequential way, that is, a short manual is available that summarise model input and output data format, program installation, price and availability, etc.

The third level is more exhaustive and presents the description of the main features of the models in hypertextual form. The user can browse the information using a set of words that are highlighted on the computer screen without following a sequential order. This technique allows a fast and direct access to the desired information.

The final and more complete level presents the complete model manuals in a hypertextual way. The manuals are stored on a CD-ROM and can be consulted directly on the computer screen even during program execution (figure 1).

2.2 Management of models with a variable structure

All the models referred to in the previous section may be thought of as basic models, i.e., as a mathematical formulation associated with a real world object that has a *flat* structure. For instance, a water reservoir can not be considered a flat domain object when it is seen as an ecosystem composed of fish, zooplankton, and phytoplankton. Conversely, when it is used to describe a simple storage of water in a more complex water management system, neglecting the ecological components, it can be characterised as a basic domain object.

A compound model, on the contrary, represents the aggregation of basic models, i.e., it defines a much more complex structure. Environmental analysts understood long ago that most environmental problems were too varied to be solvable by a fixed model structure. The development history of a river quality model like QUAL2 [10] can be seen as the progressive inclusion of a number of sub-models to better cover a variety of practical situations. The structure of these sub-models was fixed and their number for each compartment was limited. Though QUAL2 has hundreds of options, the user is requested to

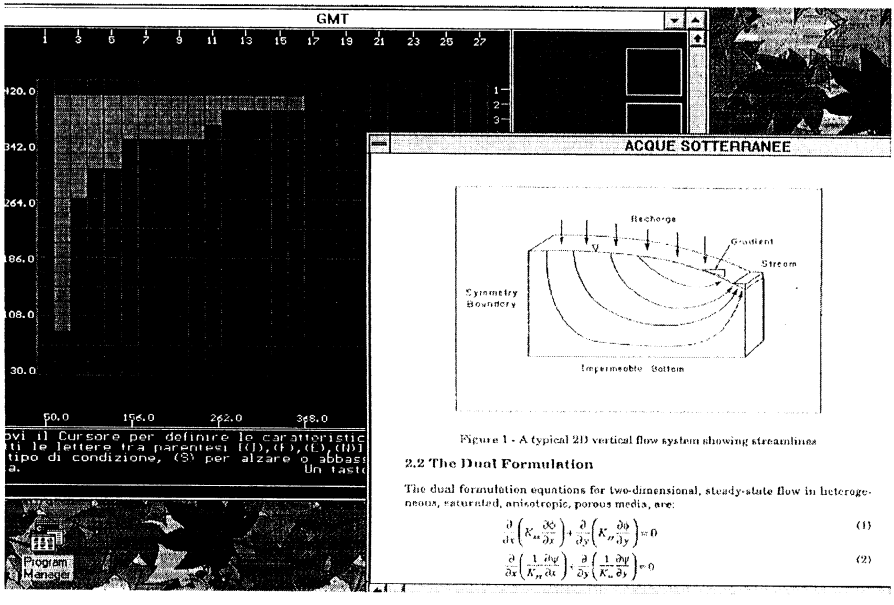

Figure 1: Accessing a specific page of the model manual, during model execution.

select in a simple menu the few that relate, for instance, to flow speed or light penetration.

Modern software engineering tools, particularly object-oriented programming, allowed to further extend the flexibility of similar programs by giving more freedom also to the overall model structure beside the choice of the components. For instance, MASAS [11] and Lakemaker [12] are recent examples of this approach to study lake eutrophication. They exploit the object-oriented approach to systematically support the model definition and execution phase. Each model, each component, each process, and even each individual variable, is represented as a separate object so that its interface with other objects is well defined. Any modification or adjustment of its data attributes may be performed without affecting the rest of the model. For instance, a specific model editor allows, in the Lakemaker package, to select the required model components, attributes, processes and parameter values (figure 2 shows the set of choices required to define the evaluation of a specific subprocess - light limiting factor).

This structure forces a useful and traditional top-down approach to model building. The user is progressively brought down from the most significant choices to the finest details with the possibility of visualising, at any time, the model structure defined up to that point.

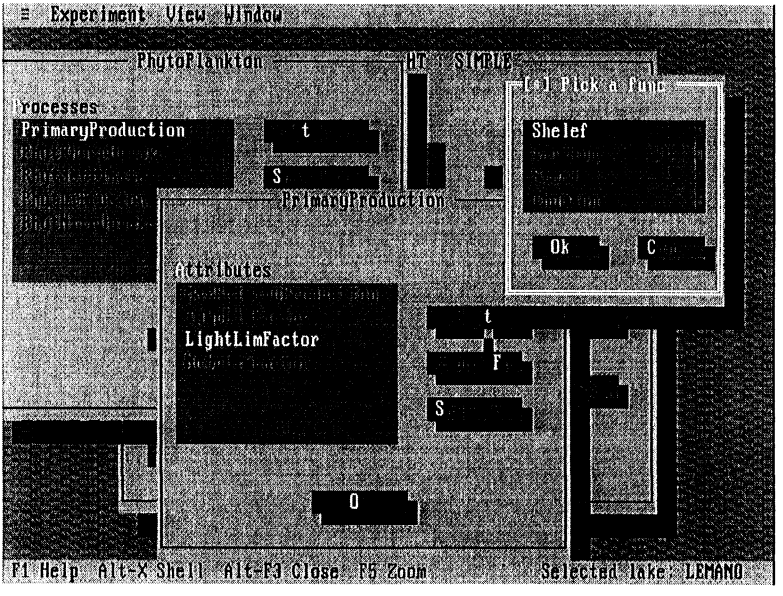

Figure 2: Selection of alternative descriptions of the same subprocess (light limiting factor).

2.3 Tools for supporting model selection

As already mentioned, assisting the user in finding a correct model is a complex task. Two modern computer tools seem particularly suitable to support this task: expert systems and hypermedia.

Expert systems have been used to support model development since much longer than hypermedia. A prototype expert system was developed to help users with the selection of the parameters needed to run the HSPF hydrologic model back in 1981 [13]. With the advent of commercial shells, many other expert systems were developed (see [14] for a survey) which were all efforts to improve the usability and effectiveness of well-known environmental models. The expert system was used to support the definition of a model with flexible structure and was thus dealing with a very narrow problem domain.

FRAME [5] has been an attempt at addressing the wider problem of selecting a legacy model in the domain of air pollution. It comprises two basic modules: a relational database and an expert system shell.

The relational database stores the possible attributes of all air quality models (more than hundred) in a large table. Each row of the table corresponds to a specific model with the last column containing the name of the file with the complete model description, its application conditions, and related bibliographical references. A consultation with FRAME starts with the description of a specific environmental situation. The user is then presented with a list of all the attributes valid for models that satisfy the first query. The

consultation proceeds by defining the characteristics of the desired models more and more specifically.

The expert system helps the user in consulting the database, which includes a reference to the set of rules that defines each attribute. Say, for instance, that the user is uncertain whether the value of the "plume" attribute is "buoyant" or "jet". He/she can invoke the expert system, which accesses the set of rules required to define these values. The user is then asked about some meteorological and physical parameters that are necessary to distinguish between the two types of plumes. Using the information provided by the user together with the rules in the knowledge base specific to that attribute, the inference engine computes the suggested value for the parameter under discussion and submits it to the user, that can use it to refine the search.

FRAME structure has the advantage of addressing a large problem domain by decomposing it into a set of small knowledge bases, which allow an easier debugging and maintenance.

Hypermedia technologies allow a more sophisticated and effective presentation of material in a given domain than the traditional, sequential printed matter. The two major features that characterise hypermedia are in fact the dominance of multimedia data (free text, pictures, graphics, video, sound, animation) over traditional formatted data and the emphasis on interactive navigation, i.e., traversing explicit connections among pieces of information.

HYPERMODEL [15] is a project based on the use of hypemedia techniques for browsing through a model base. It is based on four different entities:

- "Model" which accesses a technical description of each program in the base similar to the keywords in the database described above;
- "Theory" that provides some explanation of the theoretical basis of the models,
- "Manual" that allows to flip through the complete model documentation and, finally,
- "Documents" that shows the conditions and the results of the application of a model to specific cases.

This last entity is particularly important because it allows to insert in the system any type of information useful to describe the application of models to real cases. By looking at maps, pictures or short videos, besides traditional explanation text, the user may have a better understanding of the conditions under which the model was applied or about the simplifications that have been necessary in the modelling process; two aspects of paramount importance particularly for teaching purposes.

3 Development of new models

To further expand the generality of MMSs and to allow for the creation of really new models (not simply a new combination of existing options), developers have been forced to specify a structure for the models and their

internal components. A number of systems have been proposed based on an Operational Research perspective (decision variables, constraints, objective functions) [17, 18, 19] and others on a System Theory perspective (input, output, state variables varying in time) [20, 21].

All these packages share, at different levels, the assumption that the user is able to translate his/her problem into a mathematical formulation and then translate back a sequence of numbers in something meaningful for understanding and/or decision making.

Another feature of these systems is that the overall model must be built by connecting new or existing components with a typical bottom-up approach, quite different from the traditional environmental software, built for a user who perfectly knows the problem domain and wants to progressively refine its description. For these reasons and despite their conceptual value, these packages do not seem to represent viable tools for environmental analysts, at least in the near future. In the long term, on the contrary, some of them may become a kind of "standard" modelling language and this would justify the costs and difficulties of re-writing available models in a compatible form.

4 Integration of models from different domains

Model integration is needed when the modeller wants to intersect the competencies of various models to solve a complex problem, a common practice in environmental modelling. Model integration has been always done by hand. The modeller studies the whole problem, decomposes its domain in subdomains where he/she can apply existing models, collects the needed data (modifying them in order to comply with format specifications), and finally runs the model implementations, often feeding the output data of one model to the input of another. This activity is time consuming and very hard to replicate each time there is a change in the modelled scenario.

The purpose of model integration is to devise tools and techniques to facilitate these tasks and operations. While current research is active in studying the problems of model integration [1, 22], very few are the case studies in which real model integration took place [23].

If a model is designed from scratch, clearly the modeller has to use abstraction principles in order to represent the objects he/she has previously identified as components of the problem domain. On the contrary, when an existing model has to be integrated in a new environment, the model integrator is not free to construct his/her own representation of the problem domain: the models already contain their own (hidden) representations.

Thus, major problems when integrating legacy models are:

- mapping the legacy model to the problem domain;
- linking together legacy models.

Classification, aggregation and *inheritance* principles, inspired by object-oriented analysis and design methodologies, can be used for the purpose of integrating existing models.

The model integrator must extract the hidden representation of the problem domain contained in the model and define accordingly a set of abstract data structures (problem domain classes) which constitute its explicit representation. Then, links between the interface of the legacy model and the relative class must be established. This is not a trivial problem. The problem domain classes may present a different organisation from the data structures in the legacy model. A particular attention must be given to the mapping of similar concepts into the problem domain, in order to avoid useless duplication or unwanted simplifications.

Another difficulty that arises in integrating existing models, is to make them "communicate". There is no standard procedure. When the source code is unavailable and there is no way of writing "drivers" accessing the inner model mechanisms, the most common approach is to set up communications through data files. "Embedding" the model in the problem domain classes can overcome this problem since inter-model communications are filtered through class interfaces (data attributes and methods) which also provide a way of synchronising and transforming data items which do not have the same spatial and temporal resolution.

Figure 3 illustrates how these principles may be applied in practice [24]. Two legacy models are embedded in two separate classes (class LandSys and class StreamSys) and then integrated in a unique class Catchment.

The local data attribute of the class Catchment are re-routed to the import data attributes of the subclasses and the outputs of subclass StreamSys to the export data attributes of class Catchment. The class structure also allows to connect, via the export and import data attributes, the outputs of model LandSys to the inputs of model StreamSys.

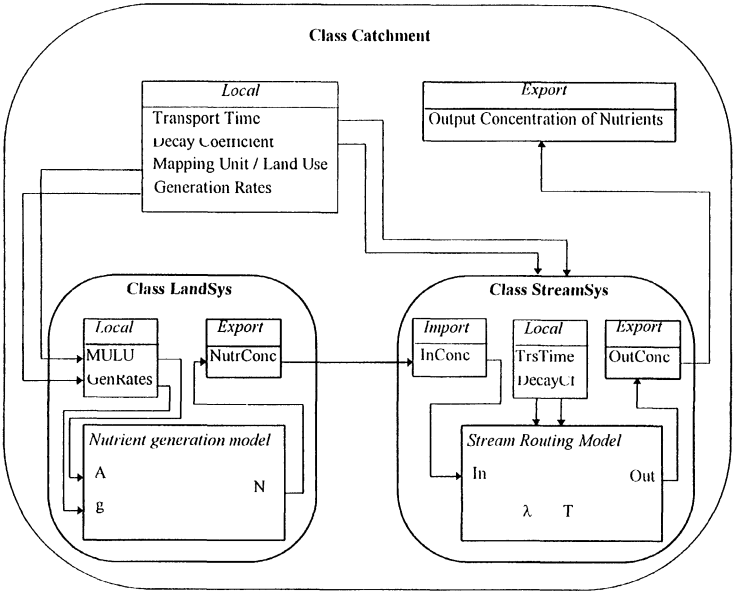

Figure 3: An example of model class for a river quality problem.

5 Concluding remarks

The development of efficient model bases and model management systems is becoming essential to support a wider utilisation of environmental models. Modern computer tools allow an improved interaction with the user and thus make this objective feasible. In this paper, we have presented some ideas and realisations of model bases for different purposes and using different computer tools. A summary of this brief survey is presented in Table 1. It shows that object-oriented techniques are probably going to have a major impact on the development of environmental modelling, while expert systems may be important whenever dealing with a sufficiently narrow knowledge domain and hypermedia when the scope of the base is more directed toward providing practical experience to the user.

Table 1: a summary of model characteristics and proposed computer tecniques.

Model characteristics / *Computer tecniques*	**Fixed structure**	**Variable structure**	**Entirely new**	**Integration of existing models**
database	very important	useful	useful	useful
expert systems	useful	important		useful
hypermedia	useful	useful		useful
object-oriented approach		very important	very important	very important

References

1. Dolk, D.R. & Kottemann J.E., Model Integration and a Theory of Models, *Decision Support Systems*, **9**, 1993.
2. Potter, W.D., Byrd, T.A., Miller, J.A. & Kochut, K.J. Extending Decision Support Systems: the Integration of Data, Knowledge, and Model Management, *Annals of Operations Research*, **38**, 1992.
3. International Ground Water Modeling Center (IGWMC), *Software Catalog*, Boulder, CO, 1995.
4. Zannetti, P. *Air Pollution Modeling.* Computational Mechanics Publications, Southampton, UK, 1990.
5. Calori, G., Colombo, F. & Finzi, G. FRAME: a Knowledge-Based Tool to Support the Choice of the Right Air Pollution Model, in *Computer Support for Environmental Impact Assessment* (eds. G. Guariso, B. Page), pp. 211-222, North Holland, Amsterdam, 1994.
6. Dodson & Associates, Inc., *HydroCD*, Huston, Texas, 1996.
7. Donlcy Technologies, *Environmental Model Directory*, 1995.
8. *Environmental Software. A strategic study of the environmental software market*, Elsevier, Amsterdam, 1992.
9. Del Furia, L. & Petrucci, M. LIVIA Project: a computer laboratory for environmental impact assessment, , in *Computer Support for Environmental Impact Assessment* (eds. G. Guariso, B. Page), pp. 311-320, North Holland, Amsterdam, 1994
10. Barnwell, T.O., Brown, L.C. & Marek W. Development of a prototype expert advisor for the enhanced stream water quality model QUAL2E. Internal Report, US EPA, 1986, Athens, GA.
11. Ulrich, M.M., Imboden, D.M. & Scwarzenbach, R.P. MASAS-A user-friendly simulation tool for modeling the fate of anthropogenic substances in lakes, *Environmental Software*, 1995, 10, 43-64.
12. Del Furia, Rizzoli, A. & Arditi, R. Lakemaker: a general object-oriented software tool for modeling the eutrophication process in lakes. *Environmental Software*, 1995, **10**, 43-64.

13.Gashing, J., Reboh, R. & Reiter, J. Development of a Knowledge-based Expert System for Water Resources problems, SRI Project 1619, 1981, SRI International, Palo Alto, CA.

14.Davis, J. R. & Guariso, G. Expert systems support for environmental decisions, *Environmental Modelling Vol. II,* ed. P. Zannetti, pp. 325-350, Computational Mechanics Publications, Southampton, 1994.

15.Guariso, G. & Tracanella, E. Hypermodel: An hypermedia guide to environmental models, in *Proc. Int. Conf. on Environment and Informatics* (Z. Harnos ed.), pp. 12-19, Budapest, 1995.

16.Muhanna, W.A., SYMMS: a Model Management System that Supports Model Reuse, Sharing, and Integration, *European Journal of Operational Research*, 1994, **72**, 214-242.

17.Bhargava, H.K. & Kimbrough S.O. Model Management: an Embedded Languages Approach, *Decision Support Systems*, **10**, 1993.

18.Dolk D.R. & Kottemann J.E, Model Integration and Modeling Languages: a Process Perspective, *Information Systems Research*, **3**, 1992.

19.Hong, S.N., Mannino, M.V. & Greenberg, B, Measurement Theoretic Representation of Large, Diverse Model Bases: the Unified Modeling Language L_U, *Decision Support Systems*, **10**, 1993.

20.Rozenblit, J.W., Jankowski, P.L., An Integrated Framework for Knowledge-Based Modeling and Simulation of Natural Systems, *Simulation*, **57**, 1991.

21.Guariso, G., Hitz, M. & Werthner, H. An Integrated Simulation and Optimization Modelling Environment for Decision Support, *Decision Support Systems*, 1996, **16**, pp. 103-117.

22.Rizzoli, A. A software architecture for model management and integration: theoretical background. *Technical Memorandum 94.10. CSIRO Division of Water Resources*. ISBN 0 643 05591 6. 1994

23.Davis, J.R, Abel, D.J., Zhou, D., Rizzoli, A. & Kilby, P. HYDRA: a Generic Design for Integrating Catchment Models, Presented at the *American Society of Civil Engineers 21st Annual Conference on Water Resources Planning and Management Division*, Denver, Co, 22-26 June 1994.

24.Rizzoli, A., Davis, J.R., Reed M. &Farley T. 1996. A DSS for catchment management. To appear in: "Environmental Software, Vol. III"; P. Zannetti (ed).