

# A Framework for Catchment Prediction Modelling

R. M. Argent<sup>1</sup>, R. A. Vertessy<sup>2</sup>, F. G. R. Watson<sup>3</sup>

1. CRC for Catchment Hydrology, Department of Civil and Environmental Engineering, The University of Melbourne
2. CRC for Catchment Hydrology, CSIRO Land and Water, Canberra
3. Institute for Earth Systems Science and Policy, California State University Monterey Bay, California.

## ABSTRACT

Models used in catchment prediction have often been developed for specific research problems or locations by individuals using software engineering practices that are now considered obsolete. The legacy of this is a range of models dealing with similar problems, using similar data input and output interpretation, but with a high diversity of operational features. Modern catchment management requires that policy development, planning and intervention be undertaken in an integrated fashion, with consideration given to physical, ecological, economic and social systems. Integrated catchment modelling offers managers some support in meeting these needs, although the range of different languages, platforms, and design approaches used in existing models means that integrated modelling can not be done by simply plugging together existing models.

A modelling framework is being developed that covers not only module design and specification, but also relevant protocols for user involvement, testing and comparison, documentation, calibration and validation, and verification and peer review. This framework will be used by the Cooperative Research Centre for Catchment Hydrology as a basis for integration of appropriate existing modules and newly developed modules into a catchment prediction toolkit. A benefit of this approach is that researchers and modellers can spend more time on core module development, and less or no time on input and output management. Other benefits include a common software operational paradigm and consistent delivery approach for a suite of catchment management related software programs. It is anticipated that adoption of this framework will support the long term design and integration of a variety of modules relevant to the prediction of catchment behaviour.

## 1 INTRODUCTION

Engineers and scientists working in hydrology and water resources were some of the earliest adopters of computer modelling techniques, resulting in a wide use of modelling in research, design, operation, prediction and forecasting, and decision support. In the areas of catchment prediction modelling, models were often developed to deal with specific research problems or locations, and were developed in times when modern software engineering practice was in its infancy. This early and continued adoption has resulted in a legacy of models that deal with similar problems and use similar data sets, but which have a high diversity of operational features such as data formatting, time stepping, spatial structure, and parameter input. Also, with few or no widely accepted protocols for documentation, peer review and authentication, testing, and calibration and validation, these important tasks were often left unrecorded or undone. Current catchment management approaches require integrated investigation and modelling of hydrological, ecological, economic and social systems, and our legacy of monolithic models do not meet these needs with sufficient flexibility and utility. Further, the complexities of temporal and spatial scaling, the issues of modelling across these scales, and the non-linearity found in many of our natural system, requires flexible approaches to catchment management modelling that are not often dealt with in other modelling disciplines.

In another way, the problems with many current models can be summed up in the phrase "water quality Model X is not too bad, but I wish I could change the routing procedure". The problem here is that the *model* is seen as an immutable combination of routines (ie a solution looking for a problem) rather than the result of a process that selects and combines appropriate routines to fit the modelling solution to the problem.

A more useful approach, and one which is supported by modern software engineering practice, is to construct the various routines as core *modules* which can be individually selected and combined to form new and appropriate applications for each different problem (Zeigler, 1995; Pressman, 2001; Ng and Yeh, 1990). When housed within a flexible modelling framework, and complimented by support modules for data handling, visualisation, documentation, and analysis, a toolkit for catchment prediction modelling can be provided. This is the future of environmental modelling!

For catchment prediction modelling, this approach is being pursued by a project within the Cooperative Research Centre for Catchment Hydrology (CRCCH) that will deliver a prototype modelling toolkit consisting of a modelling framework and a suite of modules covering a range of core hydrological applications and support functions. In developing a new framework, or adopting and adapting an existing framework, a range of technical,

usage and documentation aspects need to be addressed. This paper explores the concept of a modelling framework and the issues that are being considered by the CRCCH and other projects worldwide, and extends the concept to encompass other issues that are commonly left behind in modelling, such as data handling and visualisation, system documentation and validation, and provision of appropriate scenario comparison tools.

## 2 FUNDAMENTAL FRAMEWORK FUNCTIONS

In essence, a modelling toolkit can be thought of as a suite of tools that provide a range of functions for those undertaking modelling. These functions can include tools that:

- are core hydrological modules (eg. runoff generation, routing);
- allow new modules to be created;
- allow input data to be examined, analysed, checked, collated or modified;
- select appropriate modules to address problems at particular space and time scales;
- record or suggest appropriate parameters to be used for model runs at different scales, and
- can be turned on and off to suit the needs of different users and uses.

Underlying this toolkit is the framework - the system that allows the tools to function smoothly, seamlessly and transparently from the user perspective. Figure 1 takes a broad view of the functions that should be considered part of a modelling framework. In this, the basic concept of the framework is a system that supports the transformation by various core modules of input data into results. Using this concept, it is then possible to consider the various tools and features that a modelling framework needs to provide.

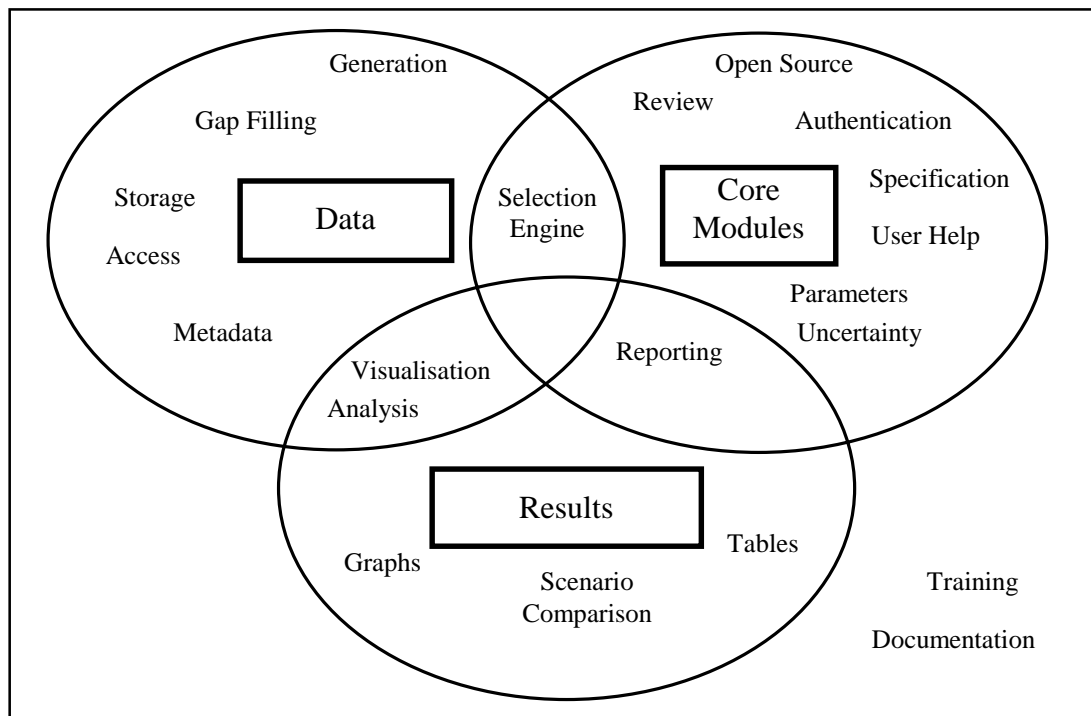


Figure 1. Functions Required of a Modelling Framework

In Figure 1, the input data are supported by tools that deal with data storage and access, metadata interpretation, and generation and gap filling to provide complete input data sets to the core modules. A suite of tools also allow analysis and visualization of both input data and modelling results. A selection engine provides a valuable tool for both selecting modules appropriate to the problem at hand, and data (and parameters) appropriate to the selected modules. Along with the core modules goes a host of tools that provide user help, track module review and authentication, deal with the notational specification of the modules, and possibly even undertake uncertainty analysis. A reporting tool provides reports of the modules and parameters used in, and results from, a modelling run. Reporting is also done through graphing and tabulation tools, and, finally, a tool or tools should be provided that undertake the comparison of the results of different modelling runs, and which provide the basics of decision support. Encompassing all of the previous are the training tools for users and developers, and the documentation tools, covering everything from module theory to application.

To establish a framework that provides the functions described above, it is necessary to consider the types of use to which this may be put, the needs of different users, and also to delve more deeply into some of the above ideas to understand the implications of the various parts on the whole system.

### 3 USERS AND USES

Another way to consider the framework functions is to consider the types of uses to which the tools provided by the framework will be put, and the needs of the system users. There are many uses for a modelling toolkit in natural resources management. For a modelling toolkit that allows for the prediction of catchment behaviour, the main uses can be defined as:

- (a) the capacity to edit, generate, examine, analyse, and visualise data. The data may be primary input data such as spatial data, economic data, or time-series biophysical data. They may also be data obtained from processing of primary data, or they may be results obtained from the running of applications;
- (b) the ability to construct new core and support modules;
- (c) the ability to combine modules to construct new applications. This may involve combining various core and support modules, and the testing and validation of the application, and
- (d) the capacity to run applications multiple times with different parameters, to test alternative management interventions.

The level of complexity involved in construction, linking and running of modules can be viewed as an embedded process, as shown in Figure 2. In this figure, each level requires a set of services to be performed at the lower level. For example, to construct applications, it is necessary to have modules available, and also to have the capacity to create new modules. This demands of the modelling framework the capacity to lump up from the level below and either add or remove features as required.

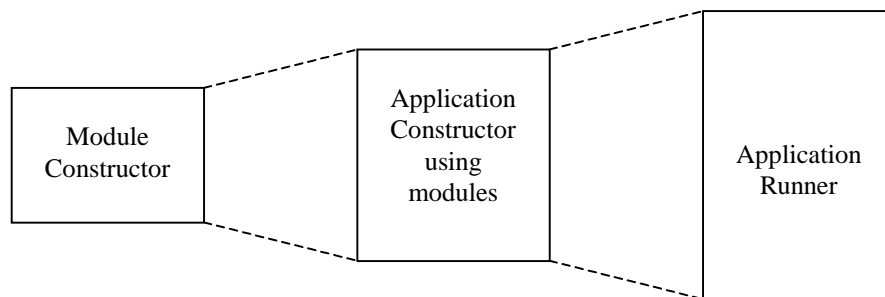


Figure 2. Levels of Use in a Hierarchical Modelling Framework

From the point of view of various users, it is envisaged that engineers and scientists working in research and management would operate at the lowest level, with access to authoring tools and the code of core modules, thereby having the capacity to create and edit routines as part of testing alternative algorithms within modules. At the next higher level, these same users, along with consultants and others wishing to craft solutions for particular problems, would have access within the framework to alternative core modules and the wherewithal to join these together, along with all the support modules for input and output manipulation, reporting and analysis. At the application running level, the focus is on a delivered product, in the hands of managers who generally have more interest in using the product than in its construction. For these users, the primary roles of the framework would be to ease the input data management burden, and to support rapid running of alternative solutions to provide decision support. For all of the users the functions allowing users to edit, generate, examine, analyse, and visualise data (ie. input data or results) would be essential.

Arising from this consideration of the users of a modelling framework, and the needs of various users, it is possible to generate the essential and desirable features of the framework. These features, discussed in the next section, form the core selection and development criteria for the CRCCH modelling toolkit development project. In considering these features, the items marked in *italics* are considered to be desirable, while those in normal font are considered essential.

#### 3.1 System Level Requirements

This details the features that pertain to the whole of the modelling framework, rather than to any of the operational levels shown in Figure 2.

- *Module based system*
- *Free of charge, or limited cost for particular third-party components*

- Open source core modules
- *Open source support modules*
- *Platform and system version independent*

### 3.2 Module Construction Level

- Modules can be edited and extended
- New module construction possible
- Defined module notation language
- Ability to wrap existing models and treat them as modules
- Safety of use of the components (errors must occur at compile time, not at run time)
- *Automatic analysis of the model possible (i.e. reverse engineering)*
- *Software must follow an agreed component standard (ActiveX, Javabeans, Delphi components)*
- *Provision of data for testing*
- *Good support from compiler and automatic code generation*

### 3.3 Application Construction Level

- Provides a framework for construction of applications by linking modules
- Individual application interfaces able to be constructed
- Use of a standard module communication architecture (eg COM or CORBA)
- Version control system
- *Support for application development in FORTRAN and VBA for Excel*

### 3.4 Application Running Level

- Contextual user documentation system
- Standard documentation (recording) of parameters and modules used in a model run
- Version control system
- *Automatic construction of application interfaces*

The previous discussion sets out an ambitious target for the scope and functionality of a modelling framework. The following section details more explicitly some of the features and issues that will be addressed in development of a framework for catchment prediction modelling.

## 4 MEETING THE REQUIREMENTS

We have identified a range of users and uses for the toolkit, and the functions that a framework might therefore have to provide for these. The following sections detail a few available options, with some further indication of the philosophy that has been adopted in our work. Issues relevant to the provision of these facilities by a modelling framework are also discussed.

- **Documentation** - Documentation is a thread running right through the development of a good modelling framework, and covers the gamut from documentation of code, recording of review and authentication processes, version control on modules, and provision of information relevant to the selection of appropriate modules for particular problem situations. At the application running level, documentation also covers recording and reporting on the data sets, modules and parameters used in various application runs, and provision of user help and training documentation. The ubiquitous nature of the HyperText Markup Language (HTML), widespread adoption of World-Wide Web technologies, the functionality of the EXtensible Markup Language (XML), and the development and adoption of the EXtensible Hypertext Markup Language (XHTML), provide some excellent indications of future directions. It is envisaged that a browser-style operating environment for a modelling toolkit would be one which would be supported technically for a long time, and which would also have strong user appeal and instant familiarity. All of the documentation needs listed above lend themselves to this style of environment.
- **Data access and storage** - Almost all core modules use data, and many use similar data, so good data handling is essential. Data are becoming increasingly available from remote sources, and tools will be required that can automatically download and manage data. Alternative approaches to handling data from remote sites also need to be addressed, as there are many options available (Argent, 2000).
- **Gap filling and data generation** - Almost all data have gaps, and tools will be required to provide intelligent infilling of data sets. Generation of stochastic data sets, and creation of spatial coverages, are essential in many catchment modelling applications, so facility will need to be provided for these.

- **Metadata standards and handling** - In catchment modelling, it is becoming an increasing requirement to provide and record metadata for data sets. Provision of framework functions that promote good use of metadata, and adherence to metadata standards, enhance the selection of data and the treatment of uncertainty in modelling.
- **Analysis, reporting, and visualisation** - A key to the usefulness of any model lies in the ability to get the results across to the users. Consequently, a range of basic analysis functions for temporal and spatial data, including statistical testing, and geospatial analysis, is essential. Reporting of model results, inputs and parameters must also be provided.
- **Specification, notation, authentication and peer review** - One of the fundamentals of modular development is expression of the module in such a way that others can understand it, using a defined specification and notation system. There are a number of notation approaches available, ranging from flowcharts and pseudocode to almost code-like notation languages, such as the Unified Modeling Language (Booch *et al.*, 1999), and a key to use of such a system in catchment modelling is finding a system that people can easily use and understand. Checking and verification of module code must also be considered.
- **Selection engine for data and modules** - One of the frustrations expressed by hydrologists arise from the plethora of models, modules, and algorithms that do what appear to be similar things. For example, in routing flows, the available options allow treatment at different spatial and temporal scales, and different levels of complexity. One of the tools thought essential to the wider operation of a modelling toolkit is an "intelligent selection engine" through the use of which users could, given the constraints of available data, select appropriate modules.
- **Scenario comparison and decision support** - In a modelling framework for catchment management, we see that it is essential to provide three services for decision support. The first of these is the ability to construct and run an application to address a particular problem situation, and this is well met by provision of the "application construction" and "application running" functions mentioned previously. The second is the ability to run applications a number of times with different parameters or even different modules, to allow examination of the consequences of alternative management interventions. This is handled in a good framework by a recording and reporting system that can retain and allow reloading of different application formulations. Thirdly, decision support is supported through the ability to report and export the results of alternative application runs to external bodies or to external analysis applications. This functionality requires the good reporting, analysis and visualisation tools mentioned previously.

Given the number and complexity of the functions described above, is it possible to provide a framework that can support it all? We think so, and the direction of development of a number of groups around the world suggests that others have the same vision. The following section briefly outlines some of the approaches that are being developed and which show promise for delivering the features specified.

## 5 DISCUSSION

The style of framework discussed has been under development in various forms and for various projects over at least the last 10 years. For some groups, the catalyst has been a similar experience to our own - wanting to re-use support modules, wishing to spend valuable coding time on core modules, and wanting to change parts of existing models. For some, it has been a question of "if we knew 10 years ago what we know now, how would we have gone about our development?".

The Dynamic Information Architecture System (DIAS) is one example of an object-oriented system that uses infrastructure classes to handle the requirements of core and support modules. DIAS uses the Common Object Request Broker Architecture (CORBA) to communicate between objects, and can utilize existing monolithic models through a registration process. An attractive feature of DIAS is the ability to construct context sensitive graphical user interfaces (Sydelko *et al.*, 1999).

The Collaborative Modelling Environment (CME) is a framework that, similarly to DIAS, allows for the creation, storage and use of hierarchical objects representing module entities. CME arose from the considerable work that went into development of the Spatial Modeling Environment (SME), used in the Patuxent and Everglades Landscape Models (Voinov *et al.*, 1999). Version 1 of CME used Tcl/Tk as the development language, with XML grammar used for modules, while Version 2 is being developed in Java.

The Modular Modeling System (MMS) has been under development for some years now, and contains a library of modules that can be selected and linked to create applications. New modules can also be created, as desired, using a builder application. The system is largely database driven with input data from text and GIS files being managed through the database and supplied to the model as required (Leavesley *et al.*, 1996).

Tarsier, a system developed partly within the CRCCH, operates as an object-oriented module housing framework that allows rapid construction of applications by either combining existing modules, written as DLLs, or by adding new modules as needed. A key feature of Tarsier is a message passing ability, whereby an application

module, such as an output graph, only updates when the information upon which it is based is updated, resulting in rapid and efficient application run times (Watson, 1998). Another development associated with the CRCCH is the Integrated Catchment Management System (ICMS). This system uses objects and classes of components for module representation, and is underlain by the Open Modelling Engine (Reed *et al.*, 1999).

These examples lie largely in the research arena. Similar approaches are embodied to a lesser or greater extent in a number of commercial and free applications, including the Surface-Water and Watershed Modeling Systems (SMS and WMS), ISIS, and the BASINS software of the United States Environment Protection Agency.

Even if one of these or other frameworks, or derivations thereof, proves to be a framework suitable for the type and style of modelling discussed, there is still no guarantee that our modelling community will be willing to move from their existing individual modelling approaches, to a more shared approach. We have no doubt that there is an openness and willingness to share amongst the modelling community, but there is less enthusiasm for adoption of higher levels of documentation and standardised notation approaches. Our strategy for tackling this is through the formation of user groups and the provision of such high levels of functionality that it proves easier to use the framework than to continue doing things in the same old way. Also, modellers move on, and new modellers, with higher software engineering skills and system expectations, are being trained in our universities today, so considerable adoption timeframes may be necessary to allow for this.

## 6 CONCLUSIONS

Development of a framework to support the module-based modelling approach discussed here will serve well the particular needs of catchment prediction modelling in the future. As indicated, a number of frameworks are currently being developed that may well be able, with some adaptation, to provide the necessary broad range of functions, and also to meet the varied needs of different framework uses. By incorporating existing models as part of testing and development, we aim to refine our framework development over the coming years to provide a working pilot that meets the needs of users at all levels. It is anticipated that adoption of this framework will support the long term design and integration of the wide variety of models relevant to the prediction of catchment behaviour.

## 7 REFERENCES

- Argent, R. M. (2000) *Integration of remote data into water resources simulation software: Now or never?* In *Environmental Software Systems. Environmental Information and Decision Support*, (Eds, Denzer, R., Swayne, D. A., Purvis, M. and Schimak, G.) Kluwer Academic, Boston, pp. 103-113.
- Booch, G., Rumbaugh, J. and Jacobson, I. (1999) *The unified modeling language user guide*, Addison-Wesley, Reading.
- Leavesley, G. H., Markstrom, S. L., Brewer, M. S. and Viger, R. J. (1996) The modular modeling system (MMS) - The physical process modeling component of a database-centred decision support system for water and power management. *Water, Air and Soil Pollution*, **90**, 303-311.
- Ng, P. A. and Yeh, R. T. (1990) *Modern Software Engineering*, van Nostrand Reinhold, New York.
- Pressman, R. S. (2001) *Software Engineering: A Practitioner's Approach*, McGraw Hill, New York.
- Reed, M., Cuddy, S. M. and Rizzoli, A. E. (1999) A framework for modelling multiple resource management issues - An open modelling approach. *Environmental Modelling & Software*, **14**, 503-509.
- Sydelko, P. J., Majerus, K. A., Dolph, J. E. and Taxon, T. N. (1999) *A dynamic object-oriented architecture approach to ecosystem modeling and simulation* In *Proceedings of the American Society of Photogrammetry and Remote Sensing Annual Conference*, Portland, OR., USA, 410-421.
- Voinov, A. A., Costanza, R., Wainger, L. A., Boumans, R. M. J., Villa, F., Maxwell, T. and Voinov, H. (1999) Patuxent landscape model: integrated ecological economic modeling of a watershed. *Environmental Modelling & Software*, **14**, 473-491.
- Watson, F.G.R., Vertessy, R.A., Grayson, R.B. and Pierce, L.L. (1998) Towards parsimony in large scale hydrological modelling - Australian and Californian experience with the Macaque model. Fall AGU meeting 1998, paper H72D-21.
- Zeigler, B. P. (1995) *Object Oriented Simulation with Hierarchical Modular Models*, B. Zeigler.