2008

Integrated Modelling Frameworks for Environmental Assessment and Decision Support

A. E. Rizzoli
IDSIA

G. Leavesley
US Geological Survey

J. C. Ascough II
USDA-ARS-NPA

R. M. Argent
Bureau of Meteorology

I. N. Athanasiadis
IDSIA

See next page for additional authors

Rizzoli, A. E.; Leavesley, G.; Ascough, J. C. II; Argent, R. M.; Athanasiadis, I. N.; Brilhante, V.; Claeys, F. H. A.; David, O.; Donatelli, M.; Gijsbers, P.; Havlik, D.; Kassahun, A.; Krause, P.; Quinn, N. W. T.; Scholten, H.; Sojda, R. S.; and Villa, F., "Integrated Modelling Frameworks for Environmental Assessment and Decision Support" (2008). *USGS Staff -- Published Research*. 196.
http://digitalcommons.unl.edu/usgsstaffpub/196

Authors

A. E. Rizzoli, G. Leavesley, J. C. Ascough II, R. M. Argent, I. N. Athanasiadis, V. Brilhante, F. H. A. Claeys, O. David, M. Donatelli, P. Gijsbers, D. Havlik, A. Kassahun, P. Krause, N. W. T. Quinn, H. Scholten, R. S. Sojda, and F. Villa

CHAPTER SEVEN

INTEGRATED MODELLING FRAMEWORKS FOR ENVIRONMENTAL ASSESSMENT AND DECISION SUPPORT

A.E. Rizzoli [a], G. Leavesley [b], J.C. Ascough II [c], R.M. Argent [d], I.N. Athanasiadis [a], V. Brilhante [e], F.H.A. Claeys [f], O. David [b], M. Donatelli [g], P. Gijsbers [h], D. Havlik [i], A. Kassahun [j], P. Krause [k], N.W.T. Quinn [l], H. Scholten [m], R.S. Sojda [n], *and* F. Villa [o]

Contents

[a] IDSIA, Galleria 2, CH-6928 Manno, Switzerland
[b] US Geological Survey, PO Box 25046, MS 412, Denver Federal Center, Lakewood, CO 80225, USA
[c] USDA-ARS-NPA, Agricultural Systems Research Unit, 2150 Centre Avenue, Bldg. D, Suite 200, Fort Collins, CO 80526, USA
[d] Bureau of Meteorology, Water Division, GPO Box 1289, Melbourne 3001, Australia
[e] Federal University of Amazonas, Computing Science Department, Av. Gen. Rodrigo Octavio J. Ramos, 3000, Aleixo, Manaus, AM 69077-000, Brazil
[f] Ghent University, Department of Applied Mathematics, Biometrics and Process Control, Coupure Links 653, B-9000 Gent, Belgium
[g] IPSC, Agri4cast Action, Joint Research Centre, Via E. Fermi, 2749, I-21027 Ispra (VA), Italy
[h] WL Delft Hydraulics, Inland Water Systems, PO Box 177, 2600 MH, Delft, The Netherlands
[i] Smart Systems Division, Austrian Research Centers GmbH, ARC, 2444 Seibersdorf, Austria
[j] Department of Social Sciences, Information Technology Group, Hollandseweg 1, Building 201 (De Leeuwenborch), 6706 KN, Wageningen, The Netherlands
[k] Institut für Geographie, Lehrstuhl Geoinformatik, Friedrich-Schiller-Universität Jena, Loebdergraben 32, D-07743 Jena, Germany
[l] Berkeley National Laboratory, University of California, 1 Cyclotron Road, Bld. 70A-3317H, Berkeley, CA 94720, USA
[m] Wageningen University, Social Sciences, Information Technology Group, Hollandseweg 1, 6706 EW, Wageningen, The Netherlands
[n] Northern Rocky Mountain Science Center, USDI, Geological Survey, 212 AJM Johnson Hall, Montana State University, Bozeman, MT 59717-3492, USA
[o] Ecoinformatics Collaboratory, Gund Institute for Ecological Economics, University of Vermont, 617 Main Street, Burlington, VT 05405, USA

7.1. INTRODUCTION

As argued in Chapter 1, modern management of environmental resources defines problems from a holistic and integrated perspective, thereby imposing strong requirements on Environmental Decision Support Systems (EDSSs) and Integrated Assessment Tools (IATs). These systems and tools tend to be increasingly complex in terms of software architecture and computational power in order to cope with the type of problems they must solve. For instance, the discipline of Integrated Assessment (IA) needs tools that are able to span a wide range of disciplines, from socio-economics to ecology to hydrology. Such tools must support a wide range of methodologies and techniques like agent-based modelling, Bayesian decision networks, optimisation, multicriteria analyses and visualisation tools, to name a few.

Sometimes EDSSs and IATs are built from scratch, often with limited resources, by non-programmers. From a software point of view, these applications are custom-made, by craftspeople rather than industrially developed by professionals. More recently, the disadvantages of this approach, which can quickly become overly expensive in terms of delivery time and resources required, have been addressed by the development of suites of software engineering tools called Environmental Integrated Modelling Frameworks (EIMFs). EIMFs have typically been designed as a response to the increasing complexity of building and delivering EDSSs and IATs.

Modelling frameworks are not a novelty per se, having made a first appearance in the management science field towards the end of the 1980s (Dolk and Kottemann, 1993; Geoffrion, 1987). The framework concept later found its way into commercial packages such as MATLAB for scientific computing, GAMS and AMPL for management science and operations research applications. Moreover, modelling and simulation tools and frameworks have been taken up on a large scale in other disciplines, and standards for developing and expanding them have been adopted. As a result, electrical circuit design toolkits and printed circuit board simulators have contributed significantly to the advancement of electronics in science and industry. The same holds for many other sectors, from the automotive industry to mechanical systems design. In contrast, no modelling framework has been universally adopted within the environmental modelling domain, and the number of environmental modelling frameworks is still growing.

A frequently asked question is: "why do we need yet another modelling frame-work?" The reasons why MATLAB (http://www.mathworks.com), MathCAD (http://www.mathsoft.com), Mathematica (http://www.wolfram.com) and simi-lar software environments are not up to the task of deploying effective and usable EDSSs are often unclear, and there is always the option of re-using an existing EIMF. Yet, this option is often disregarded, again without clear reasoning behind it.

In this chapter, we strive to address the above issues and clearly identify the essential characteristics of an EIMF. Moreover, we wish to: (1) point out the main differences among the leading EIMFs present on the scientific market; and (2) assess which characteristics justify the differences, and which characteristics are artificial and should be ignored to better facilitate interchange of knowledge and experiences in EIMF development. Finally, this chapter also advocates the development of open standards for the exchange and re-use of modelling knowledge, including data sets, models, and procedures in order to facilitate improved communication among the leading EIMFs.

7.1.1 A first definition

Definitions are tricky in that just the simple act of defining something reduces and limits its essence. Yet definitions are useful since they provide a common under-standing of the fundamental nature of things. In this chapter, we attempt to identify the essential characteristics of an EIMF while retaining the necessary flexibility to allow for the different declinations of EIMFs in practice.

Thus, a first and very general definition of an EIMF is: "a set of software li-braries, classes, components, which can be (re-)used to assemble and deliver an environmental decision support system (EDSS) or an integrated assessment tool (IAT)."

However, this definition is potentially too generic since it does not fully capture the essence of an EIMF. It also depends on adequately defining the essential func-tions provided by EDSSs and IATs. Moreover, if the EIMF itself is too generic, then the programmer and the modeller will feel more comfortable using well-assessed code development frameworks such as .NET and J2EE and mathematical mod-elling tools such as MATLAB and Mathematica, rather than taking the trouble of learning to use a new framework.

Yet, we still develop frameworks – examples are TIME (Rahman et al., 2004, 2003), OpenMI (Gregersen et al., 2007), Tornado (Claeys et al., 2006), OMS (David et al., 2002; Ascough et al., 2005), JAMS (Kralisch and Krause, 2006), and ModCom (Hillyer et al., 2003). Their proliferation leads us to ask the reason why.

7.1.2 Why do we develop new frameworks?

If we take a quick review of the most successful environmental models in different domains, e.g. MODFLOW (http://water.usgs.gov/nrp/gwsoftware/modflow2000/modflow2000.html) for groundwater modelling or the MIKE11 series of models (MIKE11, MIKE21 and MIKE Basin) for hydrodynamic modelling (http://www.dhisoftware.com/general/Product_Overview.htm), we notice that few of them

were developed using an EIMF. The Argus One Numerical Environment (http://
www.argusint.com) is as close to an EIMF that has been developed to support
popular groundwater models. This software provides graphical pre-processing and
post-processing tools available as a plug-in extension for several USGS groundwater
codes including MODFLOW as well as Arc/Info and Arcview. However, we can
also pose another interesting question: will the next generation of these types of
models be developed using more comprehensive EIMFs?

The answer is hopefully yes, but only if the EIMFs prove to be effective devel-
opment tools – otherwise they will not be used. This answer is trivial, but it also
means that we need to identify the most important features that make an EIMF a
powerful development tool.

The main reasons why we need an EIMF are: *time, money,* and *quality*.

- *Time*: we want to deliver a new application (for instance, a decision support
  system for the management of water quality in a river stretch) in a reasonably
  short period of time, e.g. within months rather than years.
- *Money*: we want to re-use what we have previously developed, and possibly re-use
  and link to what others have developed.
- *Quality*: we need to deliver results of proven quality, and for this, we need a tool
  that guides us through a proven development process.

The development of a new framework starts from these main drivers, usually
targeting a specific domain in order to solve a problem of moderate complexity.
Being a framework, it is therefore extended but evolution and growth bring in new
problems. For example the framework either becomes too difficult to maintain or
too complex to use, outlives the programming language in which it was written, or
becomes obsolete for another reason. Before this happens the framework may reach
the point where it takes less effort to build a newer, simpler to use EIMF (such
as one that is better suited to a well-defined class of problems) than to solve the
problem at hand using the existing framework. Therefore, a "generic" or "all-in-
one" EIMF seems to remain the holy grail for environmental simulation software
and this appears to be a driving force behind new EIMF development.

We believe that continually developing new EIMFs should not be the case, and
that now is the time to make better use of available resources in order to improve
existing EIMFs while controlling their growth, integrating the strengths of other
developers, and sharing common and re-usable knowledge in the form of data,
models and processes.

7.1.3 A more insightful definition

We can now assess that our previous definition of an EIMF is too generic and,
while in principle very powerful, could lead to the design and implementation of
frameworks of little re-usability because of the steepness of the learning curve of
such a framework. On the other hand, a definition that specifies, in the smallest
detail, the requirements of an EIMF would unavoidably tend to be too application-
specific, and we would end up with a one-to-one relationship between frameworks
and applications.

We think that we need to shift the development process of EIMFs in order to foster re-usability of knowledge, data and models across frameworks, thus minimising the re-coding and re-design of frameworks for fitting particular needs. In order to achieve this aim, we need a new, more insightful definition that is able to capture the essence of an EIMF. We therefore start from the name itself, which is a derivative of the keywords *environment*, *integration*, and *models*.

- *Environment* means that the framework must target the environmental domain and, even more specifically, the particular environmental sector under investigation. It must therefore provide easy access to a domain-specific body of knowledge. Yet the framework should be configurable in order to span different domains, thereby allowing real *integrated* modelling.
- Such knowledge resides both in *models* and in data, which may pertain to different sub-domains of the natural environment, and to socio-economic dimensions that are essential in IA studies.
- Finally, the knowledge must be made operational by *integration*. This means establishing causal links across domains by means of modelling and simulation.

We can therefore extend the previous definition of an EIMF by specifying in greater detail what we mean by "supporting the assembling and delivering" of environmental applications. A more robust and insightful definition of an EIMF is: "a set of software libraries, classes and components, which can be (re-)used to assemble and deliver an environmental decision support system (EDSS) or an integrated assessment tool (IAT), to support *modelling and processing* of *environmental knowledge* and to *enhance the re-usability and distribution* of such knowledge."

In the remainder of this chapter we focus on how EIMFs can support modelling and the processing of models, and how environmental knowledge, models, data and workflows can be efficiently stored, used, and exchanged across different frameworks.

7.2. A GENERIC ARCHITECTURE FOR EIMFs

Given that we have put forward sound arguments as to why an EIMF should not be too generic, it remains rather difficult to come up with an encompassing definition of the ideal architecture of such a framework. In the context of the SEAMLESS project (http://www.seamless-ip.org), for example, an architecture has been proposed that is rather generic and yet can accommodate essential EIMF components as described above. In fact, the SEAMLESS project is quite unique in trying not to develop a new EIMF from scratch, but to re-use existing ideas and components from other EIMFs. An architecture such as this is shown in Figure 7.1.

This is a layered architecture. At the bottom we find the *knowledge base* which is a semantically annotated collection of data structures, models, simulation tools, optimisation algorithms, data analysis routines and workflows. The knowledge base is populated by accessing large and heterogeneous databases. We will outline later the role of ontologies in the mediation between the databases and the knowledge base.
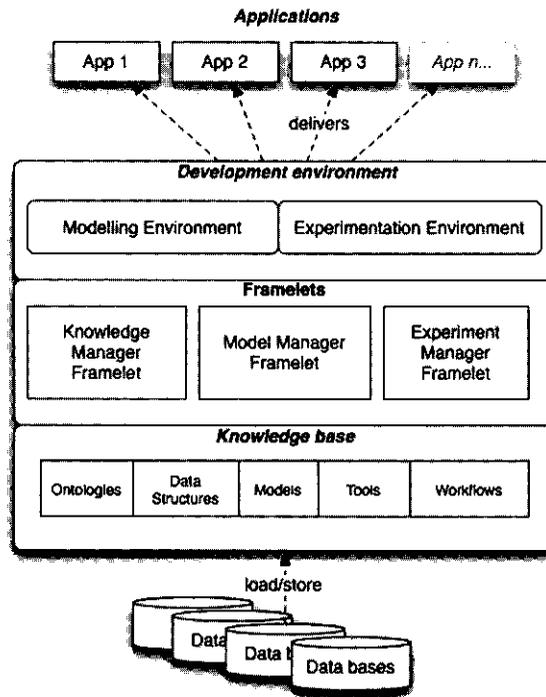
Figure 7.1 A generic architecture for an EIMF.

On top of the knowledge base, we find three "framelets" (Pasetti, 2002). Framelets are lightweight and highly specialised frameworks. The *knowledge manager* framelet provides software structures to access the knowledge base and allows the other two framelets such access to the knowledge base. The *model manager* framelet specifically targets modelling, while the *experiment manager* framelet allows the creation of workflows where tools are coupled with models to perform a number of activities, such as simulation and calibration experiments, model sensitivity analyses, output visualisation, simulation monitors, and so on. The separation between the model manager and the experiment manager allows distinguishing between the model and the operations that we perform on it. For example, in a calibration experiment the calibration algorithm is a tool that operates on the model, varying its parameters to find the best fit for instance to a given behaviour.

The framelets can then be combined into the *modelling* and *experimentation* environments. These software environments facilitate the development of end-user applications. We can think of these environments as plug-ins to existing software development frameworks such as Eclipse RCP (McAffer and Lemieux, 2005) and the NetBeans platform (Keegan et al., 2005). Of course, applications are the final product and they can be EDSSs, IATs or specific applications aimed at solving a given problem.

It is important to note that this architecture provides a blueprint that not every EIMF will follow, i.e. not all the components must be in place. The role of this

architecture is to provide a common layout to compare different EIMFs and map their architectures to a common reference.

7.2.1 A vision

We have clearly stated that the above architecture simply identifies some key elements in the structural design of an EIMF. Yet we think that the generality of the EIMF is guaranteed at the lowest level, the knowledge base, while it specialises in the upper levels. We now introduce our "heretical" vision: the most important parts of an EIMF are its components, described and annotated in the knowledge base which, if properly designed (as specified later), can stand the test of time and be used across multiple frameworks. The re-usability of knowledge base components (data, models, ontologies, workflows) across frameworks is an achievable and worthwhile goal that we should pursue.

In the next sections, we will focus on the features provided by the knowledge manager, model manager and experiment manager. In particular, we will describe software design and implementation features that will enhance the re-usability of knowledge across frameworks.

7.3. KNOWLEDGE REPRESENTATION AND MANAGEMENT

Formal knowledge representation through ontologies has been suggested as a viable solution for information and knowledge integration problems (Ludaescher et al., 2001; Villa, 2007) on the grounds that they elicit the meaning of knowledge in ways understandable by both computer systems and humans.

An ontology is a formalism for knowledge representation that comprises a vocabulary of terms representing concepts, properties and relations, knowledge domain characterisation, and formal specifications of the intended meaning of such terms (Uschold and Gruninger, 1996). As ontologies are founded on logical languages, automated reasoning can be employed in order to ensure model consistency and ontology-compliance.

The integration of models and data is the principal problem faced when building EDSSs and IATs. As we know, models and data are intrinsically related: "Science consists of confronting different descriptions of how the world works with data, using the data to arbitrate between the different descriptions, and using the best description to make additional predictions or decisions. These descriptions of how the world might work are hypotheses, and often they can be translated into quantitative predictions via models" (Hilborn and Mangel, 1997). In modelling practice, however, to consistently relate data to models is not an easy task because data, while conforming to the same paradigms and world views that inspire model conceptualisations, may not directly connect to the higher-level set of concepts necessary to describe a model. This difficulty often leads to bias and mismatches between models and supporting data sets.
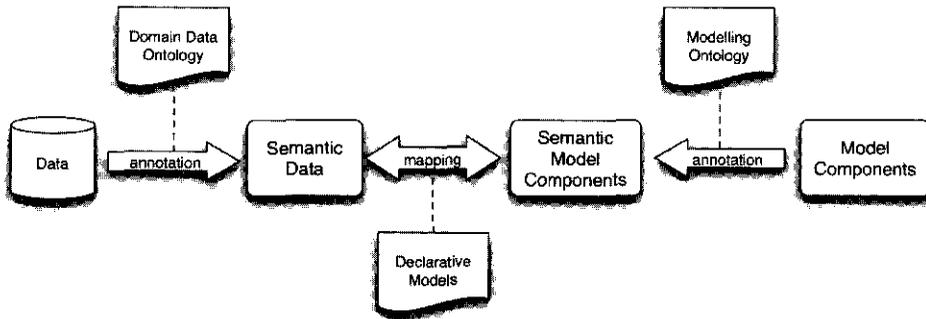
Figure 7.2 Modelling through mapping between semantic data and semantic model components.

Suppose a modelling framework exists which includes as resources (possibly distributed) ontologies, data sets, and model components. An ontology of domain data can be used to semantically annotate a data set of interest, in this way eliciting data properties and relations. Similarly, model components can be annotated through a model ontology interface, including variables, parameters and modes of operation. As a result, we would have a description of data and model components with a much narrower conceptual gap between them. Models can then be obtained by mapping the semantically annotated domain data to the semantically annotated model components. To give some examples:

- a data item annotated as being a measurement of an amount of some matter, and hence expressed in a unit of the mass dimension contained in an entity of the natural system, can be mapped to a stock concept;
- a data item annotated as being a rate of transfer of matter between entities over time can be mapped to a flow concept;
- a data table whose cells contain measurements of a data item a in relation to measurements of a data item b, annotated as a dependency relation of a with respect to b, can be mapped into a causal relationship between a and b, expressed as b being a variable in the equation that defines the variable a.

This approach can be extended to accommodate other characteristics of environmental data, such as spatiotemporal relations, economic data value changes as a result of inflation, management of complex data types, etc. Such mappings embellish existing data sources and models with advanced semantics that may lead to instantiation and linking of model components, and ultimately to declaratively defined models (Villa et al., 2006). Figure 7.2 illustrates the approach.

Of course, the realisation of such a modelling approach requires efforts to deliver – building on existing means (techniques, tools, etc.) – not only the modelling mechanisms capable of performing semantic annotation and mapping, but also the formal knowledge itself. Note that the mapping can occur in both directions and in combination: annotated data can be mapped to annotated model components and vice-versa. The former may be advantageous in modelling exercises where high-quality data are available, and the latter where suitable model components

can be identified. Either way, the mapping between semantic data and semantic model components would restrict the space of modelling solutions to plausible knowledge-level matches between conceptual model structures and data properties and relations. Modellers' expertise can further arbitrate and refine the best models out of this space of model solutions. Furthermore, the approach promotes the re-use of both data sets and model components, that is the re-use of modelling knowledge at large. Re-use of model components requires techniques to determine the right scope of the components so as to make them small enough to maximise re-usability, yet large enough to contain significant modelling knowledge.

7.3.1 Challenges for knowledge-based environmental modelling

Apart from facilitating smooth integration between data and models, the adoption of ontologies as a mediation resource has other advantages that include:

1. *The efficient definition of declarative models.* So far, declarative modelling has been concentrated on expressing model equations in a declarative way (e.g. as found in environments such as Simile, http://www.simulistics.com, and STELLA, http://www.isesystems.com, and in the Modelica framework, http://www. modelica.org). While capturing the structure of causality contained in a model remains difficult due to the simplified logics allowed by current ontology frameworks, extending model components with rich semantics for variable and parameter definitions can lead to great improvements in performing logical, dimensional and structural validation of models.
2. *The automation of scaling and unit transformations.* This is needed because data will no longer be solely vectors of numbers but will be associated with units, dimensions and spatial and temporal references that explicitly give the appropriate context to the numbers. For example, transforming an ozone concentration from 0.12 ppm into 235 $\mu g/m^3$ could be done automatically through knowledge-based tools, able to manipulate units and dimensions. Extending the coverage of the knowledge base to the conceptual aspects of space and time also allows systems to perform automatic aggregation and propagation of values over extents represented at different resolutions or with different paradigms, greatly facilitating the simulation of multiple-scale models and reducing data pre-processing overhead.
3. *Support for handling and communication of uncertainties.* Semantic annotation can greatly help the production of uncertainty records associated with measurements, suggesting or mandating the definition of known factors associated with particular methodologies, and assisting the system in propagating the calculation of uncertainties along the chain of computation when a model is executed.

A challenge that should not be underestimated is the recognition and acceptance of shared ontologies, even when the difficulties of developing, storing and maintaining large ontologies are successfully addressed. A common resistance in the scientific community to the use of ontologies is the fear of committing to a specific conceptualisation that may not fully reflect one's scientific view. Ontologies exist for eminently practical reasons, and the level of conceptual sophistication reachable

by current, first-order approaches is low enough to make such concerns relatively easy to dispel. Nevertheless, the time, training and discussion necessary to induce acceptance of such approaches in the user community is easily underestimated.

Knowledge-based computing will put modelling back in the hands of modellers: environmental modelling may become a conceptual activity, focusing on model design rather than on model implementation. Code generation and implementation of software components could be largely delegated to ontology-aware tools. In this respect, we envision the whole model lifecycle to change drastically, becoming more of a theoretical activity and less of a coding-intensive, highly engineering-oriented task. For now though, knowledge-based approaches remain a little-understood black box in the minds of most environmental scientists and engineers.

7.4. Model Engineering

While the creative activity of writing a model pertains to the area of knowledge management (as described in the previous section), here we focus on the support to model "engineering" provided by EIMFs and their model manager framelets. By engineering, we mean the set of tools and machinery that enables a modeller to transform a conceptual model, declaratively represented and semantically annotated, into computer executable code. We also include the infrastructure and software solutions that enable this model to be linked with other models, integrated in different modelling exercises, and distributed to a variety of end users and platforms.

From this viewpoint, an EIMF should: (a) allow for rapid prototyping of modelling exercises by accessing a library of models and solutions; (b) assure backward compatibility with existing (legacy) models; (c) assure interoperability with other software tools and protocols, e.g. accessing GIS tools for the purpose of building a decision support system; and (d) allow the re-use of the models developed using the EIMF within other frameworks. The ORCHESTRA network is an example of an architecture for integration and interoperability (see http://www.eu-orchestra.org), and OpenMI is another example of a framework focusing on the integration of legacy models (http://www.openmi.org).

However, note that it is practically impossible to satisfy all these constraints at the same time. Use of legacy models (point b) might impede some aspects of interoperability (point c). Moreover, the rapid development of models (point a) often assumes the use of EIMF specific libraries that imply dependencies which may be difficult to solve when re-using the model elsewhere (point d). We need to explore alternative options to find a reasonable compromise to satisfy the conflicting requirements. A solution is to approach the problem from the perspective of model representation:

1. *Run-time linkage of model engines (components) through well-defined interfaces.* Each model engine contains a model as well as its executor. Models can be implemented using different formalisms. This approach is very well suited for re-use

of legacy codes. However, it lacks maintainability and homogeneity. Linkage can be based on "push/pull" mechanisms (e.g. OpenMI) or a "bus" approach (e.g. TISC, http://www.tlk-thermo.com/tisc.html). Software agent and web-service architectures can be utilised for implementing EIMFs as virtual enterprises. Component-based modelling plays an important role in this approach.

2. *Using a declarative modelling paradigm for the description of the overall model.* Thanks to the independence of the model, expressed in a declarative fashion from its imperative implementation in code, the model is translated to an executable model description using flattening and optimisation techniques and then run by one single executor. This approach offers a high degree of homogeneity and maintainability. Moreover, the declarative representation can be semantically annotated in an ontology, allowing automated processing of the model knowledge. It is unrealistic, however, to assume that any one paradigm will ever be powerful enough to capture all modelling formalisms. In the area of physical system modelling, Modelica is an example of a high-level unified modelling paradigm. A good introduction to declarative modelling for environmental models is provided in Muetzelfeldt (2004).

3. *Translation of models implemented in different formalisms to one single, low-level common denominator.* This is a third option, an example of which is the Discrete Event System Specification (DEVS) that can be regarded as the assembly of modelling languages (Zeigler, 1990). In the DEVS approach, a translator is needed for each formalism and one executor is needed for the execution of the overall low-level model. This approach offers high potential, but has not been fully explored yet as it requires substantial development effort (de Lara and Vangheluwe, 2002).

The above approaches are not mutually exclusive and we should be able to develop EIMFs which support a mix of them. More specifically:

- If the overall model to be built is entirely situated within one particular domain (e.g. physical system modelling), one should be able to adopt Approach 2 above since this offers the highest degree of clarity, maintainability and optimisation of performance.
- If legacy models are to be used, one should be able to revert to Approach 1 above and interact with these models through standardised interfaces at run-time.

We have seen how an appropriate model representation technique can support model re-use within and across frameworks. In the next subsection we focus on the role of component-based software engineering techniques for model linking and re-use.

7.4.1 Component-based modelling

By committing to modelling frameworks as the major paradigm for model development and application, a system concept can be adopted that allows the proper assembly of models based on scientific building blocks. Such building blocks are well-defined, documented, tested and packaged. We refer to them as components, and thus advocate abandoning monolithic model development efforts in favour of

building models as a series of smaller, reusable parts. But how does general component technology translate into model development and EIMFs?

The use of components in EIMFs is not a common practice to date. Scientific building blocks in modelling frameworks do not always comply with the concept of a component for various reasons. Programming languages, overall application architectures, and legacy code requirements might constrain the design of scientific modules from a technical perspective. Such frameworks provide at least a traditional application programming interface (API) or communication protocol that can be used to implement modules. Such modules typically stay within the realm of a specific framework and therefore have limited re-use. Hence the adaptation of general component standards and their customisation for modelling has the potential for interoperability and acceptance.

Component technology focuses on the component as the primary reusable piece that allows tools to create, explore, and consume such components in a standardised way. Components are typically objects that have a predefined, documented and reusable behaviour. A scientific model component is an independent software unit that is developed for a specific scientific purpose and not for a specific model application. They are self-contained in terms of technical dependencies on other components but may rely on software outside of the framework.

As another feature, components are linkable building blocks. They can be connected at execution time using dynamic loading and linking techniques.

An EIMF that takes advantage of component technology should contain the following features:

- The EIMF allows assembly of a model from various components that can be classified into scientific, utility, control, input/output, analysis, or other types of components. They share a common structure but typically have different semantics.
- The EIMF provides flexible options to represent space and time for environmental modelling but does not constrain the user to a specific spatial or temporal discretisation concept. The EIMF might offer control components that allow for easy iteration across time and space.
- The EIMF is able to explore the component structure, a concept known as introspection, achieved using software reflection as provided by modern languages. Components might have metadata attached that specify application requirements to help domain experts with model building. Component interfaces specify data constraints or domain application requirements such as temporal and spatial constraints. Metadata structure and values could refer to a shared ontology to make linking more effective, and allow the development of tools which will rely on the type of information to implement in the EIMF, i.e. specific functionalities such as optional data quality control at run time.
- The EIMF can use components that are developed from other institutions and research groups assuming the components follow a common standard. In order to promote framework adaptation to different component and module APIs, the use of a common component standard that can be used in different EIMFs should be strongly promoted.

● Since the EIMF manages the connectivity of components, it also acts as a model linker. The model execution environment locates components dynamically, and loads and connects them for execution within the model.

Yet even if an EIMF provides all of the above features, the crucial issue of the dependency of the component on the framework has to be dealt with. A component is a piece of software that is always dependent on the specific platform and in most cases on the framework itself. For instance, a watershed model developed using the TIME framework depends on a number of TIME-specific libraries that make it difficult to re-use it 'as is' in another framework. A possible solution is to design the component as independently as possible from the framework, but we always hit the "data type" barrier. Whenever we want to exchange a data type more complex than an integer or a float, we need to rely on complex data types which have to be defined in the framework.

A possible solution is available if we allow model components to share a common interface that can be created from a public ontology, including the selection of quantities and attributes to develop data structures. Data structures can be saved as domain objects in a knowledge base, and can be further extracted in RTF or XML formats to generate domain object code for specific implementations (Athanasiadis et al., 2006). This approach goes hand-in-hand with declarative modelling where the solution to a modelling problem in various implementations is via code generation, targeting, and optimisation with respect to a specific EIMF. In this case, the constraint for re-usability is given by the appropriateness of model use in a specific context whereby possible technical issues (e.g. language, platform, EIMF functionalities) can be overcome.

Regardless of the choice of developing framework-specific or intrinsically re-usable components, there is a basic choice that must be carefully evaluated beforehand. This choice is related, in general terms, to the framework as a flexible modelling environment for building complex models (i.e. model linking), but also to the framework as an efficient engine for calibration and simulation of model components (i.e. model execution).

7.4.2 Distributed modelling

Taking the model component concept further, we envision model component interfaces published on the Internet and accessible for re-use, either to a specific community or the public in general. In a "marketplace-like" open source environment, models of fine granularity could be registered and their services made available in a collaborative fashion. In a service-oriented architecture, an EIMF could eventually become a composite service and end-users will be able to build and develop from the existing services available. Environmental data will operate as a *virtual resource* shared among peers, instead of a scarce resource for which peers strive (Athanasiadis, 2007). Employing software agents, web services or grid technology for realising a service-oriented approach, the members of a "virtual modelling marketplace" will be able to construct scientific workflows for combining original data sources with environmental models and reporting tools, all available as services. Although such a vision seems very promising, a realistic implementation is hindered

by the lack of standards for exchanging environmental data. The development and the wide adoption of community standards is a prerequisite for achieving smooth information flow within a virtual modelling marketplace. Developments similar to ebXML (http://www.ebxml.org), which is a widely adopted standard in the electronic business sector, are required for maximising the interoperability and re-usability potential of a web-based, open environment for modelling.

7.5. DRIVING AND SUPPORTING THE MODELLING PROCESS

Modelling is an iterative process (see Chapter 2) during which several activities need to be accomplished including pre-modelling tasks such as problem description and requirements analysis. Common problems encountered during modelling include inadequate project setup, insufficient or inappropriate use of methods, and lack of documentation and transparency. In multidisciplinary projects, modellers from different disciplines often do not always understand one another because of different methodological approaches and developments. To improve the quality and credibility of modelling results and model-based decision support in general, an EIMF needs to support conceptualisations that provide guidance on the modelling workflow. Such guidance can be effectively delivered only if we acknowledge the clear distinction between a model and its experimental frame.

7.5.1 The experimental frame

One essential requirement for re-usability and maintainability in complex EDSSs is the strict separation between the model and its experimental frame. By model we mean the (mathematical or other) representation of the physical system under study. The experimental frame is the environment in which the model resides. It is the experimental frame's responsibility to provide to the model the input it needs, and to accept and further process model output that is generated. Many well-known legacy "models" are actually codes in which the representation of the physical system and its experimental frame are fully intertwined. Options for re-use are therefore often limited in these cases.

In order to tackle the complexity of models, techniques such as hierarchical decomposition and object-orientation are commonplace. These techniques are especially relevant since in many cases a set of basic, atomic models (e.g. unit processes) can be identified from which other more complex models can be derived through coupling or inheritance.

Unfortunately it is not always understood that the experimental frame can also be regarded as a model in its own right, to which hierarchical decomposition and object-orientation can equally be applied. One can indeed identify a set of basic operations that apply to nearly all models, and from which other more complex types of model processing can be derived. A non-exhaustive list of the basic types of model processing (sometimes also referred to as "virtual experimentation," (Claeys et al., 2006)) include:

- dynamic simulation and steady-state simulation;
- optimisation with regard to various objectives (e.g. parameter estimation) in conjunction with confidence information analysis;
- local and global sensitivity analysis;
- scenario evaluation;
- risk analysis; and
- optimal experiment design.

Only a few EIMFs exist in which a full set of basic model processing operations is available. Even fewer allow for compound operations to be constructed from these in a flexible manner. However, in order to facilitate the process of building complex EDSSs, there is a need for tools that support hierarchical and object-oriented modelling of experimental frames – this is the role of the experiment manager framelet shown in Figure 7.2.

Various authors have aimed their work at providing tools and methodologies to support the modelling process. Jakeman et al. (2006) provide a list of ten steps to support a disciplined model development approach. Castelletti and Soncini-Sessa (2006) show how a participatory and integrated planning procedure can be formally specified in a sound methodological approach, also supported by a software tool. Giupponi (2007) shows how the DPSIR approach can be used in the context of a DSS for implementing the EU Water Framework Directive (WFD).

Business process modelling standards in general provide conceptualisations that describe the type of information required by workflow systems, but these standards lack the support of formal semantics and ontological structures to represent modelling methods, activities and other related information. The HarmoniQuA quality assurance framework tries to address some of these issues by proving quality assurance guidelines, a knowledge base and an associated modelling support tool (MoST) (Refsgaard et al., 2005).

In conclusion, an EIMF should support features that allow process management tools to be integrated with the framework so that the modelling process can be properly managed, transparent and quality-controlled.

7.6. CONCLUSIONS

In this chapter we have described the modern concept of the EIMF based on past experience, current endeavours and future plans. The existence and adoption of today's EIMFs give us hope for future improvements as we continue to research and develop, apply and experiment, and think and test. EIMFs such as JAMS, TIME, OMS, OpenMI, Tornado and ModCom show how far we have come and some of the problems we still face. Many of these frameworks share component-based approaches, use object-oriented techniques, conform to sets of design patterns and fit, in varying ways, the template offered by the generic EIMF architecture in Figure 7.1. However, they also differ fundamentally in their technological basis, such as use of NetBeans or .NET, their structure of classes and methods, data handling approaches, operational sequencing, and even in the way that they appear to a user.

These differences arise not through any particular disagreements or differences of opinions amongst developers, but through the origins and evolution of the frameworks, the institutional context of development, and the pressing needs in application. Almost all existing EIMFs have previous lives, and have been built and rebuilt to meet specific (and increasingly generic) needs. They often represent institutional knowledge that must be retained in the next generation, lest such knowledge be lost and the new EIMF end in failure. Given this evolution, it is wise to consider carefully new directions and the promise of converging on the somewhat "heretical" vision espoused in this paper. At the same time, user needs must be met so we cannot forget the fundamental importance of providing helpful, familiar and intuitive user interfaces, support for legacy systems, and seamless access to current and expanding data sources that may be distributed, disconnected, incomplete, incompatible and inconsistent.

In this chapter, we suggest that a promising trend for knowledge representation is the use of ontologies that have the capacity to elicit the meaning of knowledge in a manner that is logical, consistent and understandable by computers and the knowledge worker community. This new path in knowledge-based computing will support retention of institutional knowledge, while putting modelling back in the hands of modellers. Environmental modelling will then become a conceptual activity, focusing on model design rather than model implementation, with code generation being delegated to some degree to ontology-aware tools. In this respect, we envision the whole model lifecycle to change drastically, becoming more of a theoretical activity and less of a coding-intensive, highly engineering-oriented task. Environmental science and technology is multifaceted and individual disciplines are unequal in their reliance upon models and computer-science-aided approaches. Hence we need to retain flexibility and encourage creativity in the knowledge-based approaches we develop. There has never been a better time to reflect on some of these past successes to contemplate the next paradigm leap in computer-aided decision support.

REFERENCES

Athanasiadis, I.N., Rizzoli, A.E., Donatelli, M., Carlini, L., 2006. Enriching software model interfaces using ontology-based tools. In: Voinov, A., Jakeman, A.J., Rizzoli, A.E. (Eds.), Proceedings of the iEMSs Third Biennial Meeting: "Summit on Environmental Modelling and Software". International Environmental Modelling and Software Society, Burlington, USA, July 2006. ISBN 1-4243-0852-6. ISBN 978-1-4243-0852-1. CD ROM. Internet: http://www.iemss.org/iemss2006/sessions/all.html.

Athanasiadis, I.N., 2007. Towards a virtual enterprise architecture for the environmental sector. In: Protogeros, N. (Ed.), Agent and Web Service Technologies in Virtual Enterprises. Idea Group Inc., pp. 256–266.

Ascough II, J.C., Flanagan, D.C., David, O., Ahuja, L.R., 2005. Assessing the potential of the Object Modeling System (OMS) for erosion prediction modeling. In: Proceedings of the 2005 ASAE Annual International Meeting, ASAE Paper 052011. Tampa, Florida, July 17–20.

Castelletti, A., Soncini-Sessa, R., 2006. A procedural approach to strengthening integration and participation in water resource planning. Environmental Modelling and Software 21 (10), 1455–1470.

Claeys, F., De Pauw, D., Benedetti, L., Nopens, I., Vanrolleghem, P.A., 2006. Tornado: A versatile efficient modelling and virtual experimentation kernel for water quality systems. In: Voinov, A., Jakeman, A.J., Rizzoli, A.E. (Eds.), Proceedings of the iEMSs Third Biennial Meeting: "Summit on Environmental Modelling and Software". International Environmental Modelling and Software Society, Burlington, USA, July 2006. ISBN 1-4243-0852-6, ISBN 978-1-4243-0852-1. CD ROM. Internet: http://www.iemss.org/iemss2006/sessions/all.html.

David, O., Markstrom, S.L., Rojas, K.W., Ahuja, L.R., Schneider, I.W., 2002. The Object Modeling System. In: Ahuja, L., Ma, L., Howell, T.A. (Eds.), Agricultural System Models in Field Research and Technology Transfer. Lewis Publishers, CRC Press LLC, pp. 317–331.

de Lara, J., Vangheluwe, H., 2002. AToM3: A tool for multi-formalism and meta-modelling. In: European Joint Conference on Theory And Practice of Software (ETAPS), Fundamental Approaches to Software Engineering (FASE). In: Lecture Notes in Computer Science, vol. 2306. Springer-Verlag, Grenoble, France, pp. 174–188.

Dolk, D.R., Kottemann, J.E., 1993. Model integration and a theory of models. Decision Support Systems 9 (1), 51–63.

Geoffrion, A.M., 1987. An introduction to structured modelling. Management Science 33 (5), 547–588.

Giupponi, C., 2007. Decision Support Systems for implementing the European Water Framework Directive: The MULINO approach. Environmental Modelling and Software 22 (2), 248–258.

Gregersen, J.B., Gijsbers, P., Westen, S.J.P., 2007. OpenMI: Open modelling interface. Journal of Hydroinformatics 9, 175–191.

Hilborn, R., Mangel, M., 1997. The Ecological Detective. Princeton University Press.

Hillyer, C., Bolte, J., van Evert, F., Lamaker, A., 2003. The ModCom modular simulation system. European Journal of Agronomy 18 (3–4), 333–343.

Jakeman, A.J., Letcher, R.A., Norton, J.P., 2006. Ten iterative steps in development and evaluation of environmental models. Environmental Modelling and Software 21 (5), 602–614.

Keegan, P., Champenois, L., Crawley, G., Hunt, C., Webster, C., 2005. NetBeans 5.0 IDE Field Guide: Developing Desktop, Web, Enterprise, and Mobile Applications. Prentice Hall.

Kralisch, S., Krause, P., 2006. JAMS—A Framework for Natural Resource Model Development and Application. In: Voinov, A., Jakeman, A.J., Rizzoli, A.E. (Eds.), Proceedings of the iEMSs Third Biennial Meeting: "Summit on Environmental Modelling and Software". International Environmental Modelling and Software Society, Burlington, USA, July 2006. ISBN 1-4243-0852-6, ISBN 978-1-4243-0852-1. CD ROM. Internet: http://www.iemss.org/iemss2006/sessions/all.html.

Ludaescher, B., Gupta, A., Martone, E.M., 2001. Model-based mediation with domain maps. In: 17th Intl. Conference on Data Engineering (ICDE). Heidelberg, Germany.

McAffer, J., Lemieux, J., 2005. Eclipse Rich Client Platform: Designing, Coding, and Packaging Java Applications. Addison-Wesley Professional.

Muetzelfeldt, R.I., 2004. Declarative Modelling in Ecological and Environmental Research. European Commission Directorate-General for Research, Position Paper No. EUR 20918. European Commission, Brussels, Belgium.

Pasetti, A., 2002. Software Frameworks and Embedded Control Systems. Lecture Notes in Computer Science, vol. 2231. Springer-Verlag, Berlin.

Rahman, J.M., Seaton, S.P., Perraud, J.-M., Hotham, H., Verrelli, D.I., Coleman, J.R., 2003. It's TIME for a new environmental modelling framework. In: Post, D.A. (Ed.), MODSIM 2003 International Congress on Modelling and Simulation. Modelling and Simulation Society of Australia and New Zealand Inc., Townsville, pp. 1727–1732.

Rahman, J.M., Seaton, S.P., Cuddy, S.M., 2004. Making frameworks more useable: Using model introspection and metadata to develop model processing tools. Environmental Modelling and Software 19 (3), 275–284.

Refsgaard, J.C., Henriksen, H.J., Harrar, B., Scholten, H., Kassahun, A., 2005. Quality assurance in model based water management—Review of existing practice and outline of new approaches. Environmental Modelling and Software 20 (10), 1201–1215.

Uschold, M., Gruninger, M., 1996. Ontologies: Principles, methods and applications. The Knowledge Engineering Review 11 (2), 93–136.

Villa, F., 2007. A semantic framework and software design to enable the transparent integration, reorganization and discovery of natural systems knowledge. Journal of Intelligent Information Systems 29, 79–96.

Villa, F., Donatelli, M., Rizzoli, A., Krause, P., Kralisch, S., van Evert, F.K., 2006. Declarative modelling for architecture independence and data/model integration: A case study. In: Voinov, A., Jakeman, A.J., Rizzoli, A.E. (Eds.), Proceedings of the iEMSs Third Biennial Meeting: "Summit on Environmental Modelling and Software". International Environmental Modelling and Software Society, Burlington, USA, July 2006. ISBN 1-4243-0852-6, ISBN 978-1-4243-0852-1. CD ROM. Internet: http://www.iemss.org/iemss2006/sessions/all.html.

Zeigler, B.P., 1990. Object Oriented Simulation with Hierarchical, Modular Models. Academic Press.