A Framework for Developing Environmental Models

Aitkenhead, M.J.. and Aalders, A.H.

The Macaulay Institute, Craigiebuckler, Aberdeen AB15 8QH, E-Mail: m.aitkenhead@macaulay.ac.uk

Keywords: *Modeling; environment; knowledge framework; complex systems; neural network.*

EXTENDED ABSTRACT

Environmental modelling often involves the integration of multiple datasets, including both quantitative and qualitative data, into an implementation that is required to operate over many different scales, both spatial and temporal. This implementation can also contain variables that are spatially discrete, global or descriptive of agents within the system. As such, the development of meaningful environmental models which deal with even small systems can be extremely difficult. Here, we present a methodology, implemented within first-version software still undergoing development, which allows a researcher to produce complex knowledge frameworks and to use this framework to build a model of a complex system. The method used is based on the identification of components within the system, description of their characteristics (scale, type, categories etc.) and development of a network of relationships that describe how individual system components interact with one another. Existing models representing the relationships between components can be implemented within the system. Additionally, integration of datasets into the model using a neural network/simulated annealing component separate from the Meta software, but which is intended to be included within the package in later versions, allows relationships that are not mathematically described to be included in a model. The suite of software packages is applied to a case study involving a river catchment in NE Scotland, and provides an early indication of the potential of this modelling package design.

The manner in which modelling is carried out by the Meta package is designed specifically for use by researchers who have little or no experience working with programming languages or with commercial model-building software. As such, it is designed around a graphical interface that allows drag-and-drop and toolbar-driven model development and implementation, with the actual model description and the method of running the models hidden from the user. There is a limit to the flexibility of the system when compared with more sophisticated, and therefore more complex model software packages. Meta is not designed to replace these, but rather is intended to provide a method by which users can implement their knowledge into working, visually intuitive simulations. The software is useful for first-look examinations of system models and for exploring the relationships between system components, but it does not have the advanced capabilities of other packages, i.e. numerical approximation or statistical analysis of model results.

Figure 1 shows an implementation of a relatively simple non-spatial model, with the model components having been arranged on-screen using the user interface to produce a visually intuitive layout. Lines between variables are either all white (each of the variables connected by the line influence one another directly) or half black, half white (the variable at the white end affects the variable at the black end directly).

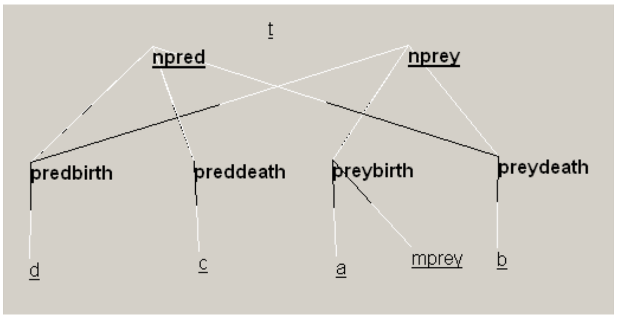


Figure 1. Model design showing predator/prey dynamics

As can be seen from Figure 1, Meta allows the user to manipulate the model representation, while at the same time automatically showing the relationships between components of the modelled system. Further choices open to the user include manipulating variable names and characteristics, and changing the relationships between variables, i.e. by adding or removing a variable from a relationship, or by adjusting the equation or rule which defines the relationship itself.

1. INTRODUCTION

Existing model-building software is commonly designed to satisfy a particular purpose (e.g. GeneFold, which is used to identify protein folding, or COGENT, a graphical system for developing cognitive models). There are few shareware or freeware applications that allow the user to develop models without extensive training or experience in computer programming. Many potential users of modelling software are discouraged from applying their personal expertise to the development of models, while experts in a particular field often develop a specific type of modelling system that is too narrow in scope for people outside their discipline to find of use. This situation therefore often prevents researchers from developing frameworks of knowledge and testing theories rigorously.

Other general model-development software does exist, and comes in a wide range of capabilities and prices. Examples include GoldSim, a graphical interface platform for systems modelling, and Matlab. There is also a large array of software libraries available for different programming languages, allowing software developers to plug in premade modelling components. Examples include HighMAST, an object-oriented class library built using Microsoft .NET technology (Highpoint Software Systems), and TomasWeb, a simulation library for Delphi users. Each of these examples, along with the vast majority of similar applications available, is more sophisticated than the software package being demonstrated here. The rationale behind Meta, however, is that sophistication is not always necessary, especially when it comes at great cost, either financially or in terms of the amount of time and effort required to learn how to use the system.

The existence of a user-friendly, intuitive and free model-building software package would remove these restrictions, and would provide the additional benefit of allowing researchers to describe their work to others in an easily visualised format. The software package demonstrated here, entitled Meta, is intended to satisfy model-building requirements from the point of view of scientists working with either closed-system or spatially explicit study systems. The intention is not to provide a complete, one-size-fits-all package that can be used by anyone, for any purpose; it is suspected that due to the number of different potential users of such a system, and the range of subjects that it would need to be able to be applied to, that such a package would be impossible to produce; certainly it would have to be updated and revised constantly to deal with new requirements. Nevertheless, there has been an observed need for researchers working with complex systems (particularly environmental scientists) to implement their knowledge, both in order to provide a basis for interdisciplinary work and to allow scenario modelling and theory testing.

2. METHODS

2.1. Graphical user interface

In order to allow the user to visualise and comprehend the model that they are building, any software package must have a graphical user interface capable of satisfying the following requirements:

- The ability to manipulate a visual representation of the model to demonstrate relationships between variables within a system
- Control over the font size and other graphical parameters in order to optimise the model display
- Intuitive dialog between the user and the software allowing models to be built, altered and implemented
- When running a model, an indication must be given of the model status in terms of both run duration and the state of model parameters

In order to satisfy the above requirements, Meta is designed in such a way that the user is presented with a range of options that may vary according to the situation. For example, if the user is in the process of building a model, Meta will provide dialog boxes allowing addition and removal of parameters, and the ability to manipulate the model as it currently stands on the screen. If the user has completed a model and wishes to run it, then a different set of choices allowing the display parameters of the running model to be adjusted will be given. The aim is to provide a clear and intuitive interface that can be taken up and used with no prior training and the use of which is obvious, while at the same time providing a help file describing the actions available to the user and the consequences of each action.

2.2. User options – variables

Several options are available to the user which allow adjustments to be made to the model components. These include renaming, redefining (in terms of type, i.e. whether they are continuous or discrete), and relationship with other variables within the model. Each of these choices is easy to implement using the user interface, and adjustments made to one variables within the model are automatically incorporated into other model components that the changed variables are related to in any way.

2.3. User options – relationships

Relationships between model variables are key to the ability to develop and implement models, and as such it is necessary to be able to define these relationships as suitably as possible. The user can change relationships after they have been defined, both in terms of the mathematical parameters within an equation and in terms of the variables that are entered within the relationship. This last capability allows the user to redefine a model once new information is made available, and to experiment with different theories of how a system operates.

2.4. User options – model display

The user has been given control over several aspects of the model display parameters, including text font and line thickness. Additionally, the user is able to drag and drop variables around the screen in order to develop more visually interesting or intuitive model diagrams. When a model is saved following these changes, the display parameters are also recorded so that when the model is next opened, it will be displayed identically to its appearance when it was last saved.

2.5. Model expression

When the user saves a model, it is stored as a text file using a straightforward encoding system that allows the model to then be opened at a later date and edited or run. The format for storing a model involves stating the names and attributes of each variable, including the graphic display settings given by the user for representing the model, and a definition of each relationship between variables. Additional information is given about the location of baseline data within the memory file system and of the file used for storing results of model runs, if available. The saved model is not translated or encoded in any way that makes it impossible to read directly using a text editor, and theoretically this could allow an experienced user to describe a complete model using any word processing package, but in practise this is sufficiently difficult to do that developing the model using Meta is much easier.

2.6. Starting a model

In addition to the definitions of variables and their relationships within a model, it is necessary to have a set of values from which to initialise the model, and also to have a location where the results of the model implementation can be stored. Both of these file locations are obtained from the user through automatic dialog boxes, with the additional option available to the user of selecting the frequency with which the model values are saved. This can be useful if the model is to run for an extremely large number of time cycles, with data only being saved (for example) every ten, hundred or thousand cycles.

2.7. Individual models - characteristics

When developing individual models (those that describe the internal workings of a single system, rather than spatially explicit models), the user can make this specification at the beginning of the model development process. Selecting this option means that Meta then implements a further set of choices which reflect the development of an individual model.

2.8. Spatial models – characteristics

In the situation that the user wishes to develop a spatially explicit model, informing Meta of this choice results in a further set of questions which allow the model to be initialised. The requirements for a spatially explicit model include information about the dimensions of the model, the scale of individual cells and further specification of variables within each cell which allow for information to be transmitted between cells, such as the time period in seconds over which each action takes place. The ways in which this information can be transmitted includes only material flow at the moment, with the intention to implement state information changes in a later version.

2.9. Implementing a model

The step-by-step development and application of a model includes the following processes:

1. User taskbar selection of either starting a new model
2. Declaration of model start and stop time
3. Primary variable declaration (this can be any variable within the system, but an appropriate choice would be a variable of particular interest)
4. Definition of primary variable type (i.e. continuous, discrete, spatial or system-wide, time step update)
5. Specification of variables influencing primary variable, and declaration of whether influence

is spatial in nature (i.e. does value of variable A in neighbouring cell affect variable B?)
6. Repeated iteration of steps 3-5 on every new variable, until all relationships are declared
7. Iterated definition of relationships specifying each variable (i.e. mathematical relationship for continuous variable, conditions of state change for discrete variable)
8. File locations or value definition for variables that are not influenced by any other variable
9. Specification of starting values or location of baseline data files for each variable, and of file location for model results
10. Option of saving model or graphically manipulating model diagram (this can be done at any time)
11. Model run, variable display on-screen

When running a developed model, the system first determines the most appropriate time period for each model cycle. This is commonly taken to be the smallest time step specified by the user for the updating of a certain variable, but if there are a range of timescales within the model for which the smallest value is not a common denominator, then the highest common denominator timestep is sought. For relationships that have been specified as acting over a timestep different from this 'core' time period, the effects of the relationship are adjusted accordingly to fit onto the core timestep value (i.e. with a core timestep of 2 seconds, a variable that is updated every 10 seconds will actually be updated on a 2-second basis, with the values within relationships affecting that variable divided by 5). Obviously, using this highest common denominator can result in vastly increased computational requirements for the model if there are a lot of 'slow' relationships and some 'fast' ones, as the 'slow' relationships will be called equally as often as the 'fast' ones.

2.10. Case study

A case study has been selected that allows demonstration of the potential of the Meta package. The Ythan catchment in NE Scotland contains a wide range of land cover and land use types, and covers a region including hills, wide valleys and coastal areas. The catchment is important to migratory seabirds and other wildlife, and the estuary at Forvie Sands is a nature reserve while another section has been granted SSSI (Site of Special Scientific Significance) status, including the area containing the single largest sand dune in the United Kingdom. The coastal area is commonly enjoyed by walkers and naturalists, while at the same time being threatened by high levels of nitrogen runoff from arable land further up the catchment. As such, the Ythan catchment is extremely important both commercially and environmentally, with many competing interests in the area.

One of the results of the Ythan catchment's importance is that it has been extensively studied by environmental and land use scientists for many years. This has resulted in a wealth of information about the soils, land cover, wildlife and other aspects of the catchment, providing a useful source of data for modellers. Also available is information about the level of nitrification of the water arriving at the Forvie Sands estuary, which will be used as a comparison with the predictive model.

Pools of nitrogen that are considered in the model include atmospheric, nitrate, ammonium and organic nitrogen. The processes by which these pools are added to or reduced include the following:

- Denitrification into the atmosphere from the soil
- Dry & wet deposition from the atmosphere into the soil
- Fixation by vegetation from the atmosphere
- Release into the atmosphere by burning
- Plant & animal additions to the soil
- Ammonium volatilisation from the soil
- Mineralisation of organic nitrogen into ammonium (and the reverse)
- Nitrification of ammonium into nitrate
- Leaching from the soil
- Plant uptake from the soil
- Organic nitrogen addition to the soil as plant fertiliser
- Water transport

Each of these processes takes place as a function of the vegetation, soil, climate and land use at a particular location. A simplified model of each process has been developed showing how much nitrogen moves from one pool to another, and from one cell in the model to another. Values associated with each process were obtained from several sources, including Batey (1982), Andren *et al.* (1991), Sarjala & Potila (2005) & Chapman *et al.* (2001).

Datasets from which relationships were to be derived using neural network modelling methods were not possible to utilise using Meta alone, as this process has not yet been implemented within the package. Instead, specific additional neural network and simulated annealing programs were required in order to produce these relationships, followed by translation of the neural networks into more simple equations of similar function. This highlighted additional work that is required in a later version of the software, as is described later.

3. RESULTS

3.1. Data acquisition

Although, as stated earlier, the Ythan catchment has been extensively studied over a period of years, and by a wide of scientific experts, there are still issues with the acquisition, interpretation and implementation of the data within the model. Variables must be used within the developing model that relate to the available data, and where there is no information about a particular process then proxies must be investigated and applied. Data that was available included the following:

- Land Cover of Scotland 1988 (LCS88), available at a 50m resolution. This dataset contains 127 individual classes and a total of over 1300 mosaic classes, and is biased towards seminatural vegetation (i.e. no distinction within arable or urban classes, but differentiation between dry heather moorland and wet heather moorland, different types of grassland, etc.). The LCS88 was used to provide a baseline for land cover over the study area with the dominant class within each 50m 'pixel' of the model used.
- Digital elevation model. The information was used to provide values for elevation, slope angle and slope direction, and also to provide proxy information for temperature.
- Daily rainfall, using the daily rainfall measurements taken at a weather monitoring station sited 50km away. There was no rainfall data available from within the Ythan catchment, and so the values given could only be used as an approximation.

3.2. Modelling the Ythan

Several layers of information were used to produce a model of the Ythan catchment, including soils, topographic, land cover and climate data. The soils data, obtained from the Soil Survey of Scotland (Macaulay Institute, 1982), was used to develop a map of soil type containing six categories (podzolic soils, brown earths, sand, peat, alluvium and gleys). A digital elevation model was used to provide values for elevation, slope and aspect. A land cover map (the Land Cover of Scotland 1988 database, Macaulay Institute, 1993) was used to develop a map of land cover containing eight classes (arable, forest, water, built-up areas, grassland, heather and moorland). Each of these spatial datasets gave information at a resolution of 50m. Climatic data gave hourly temperature and rainfall data for a year, obtained at a weather monitoring station approximately 50 km away. Figure 2 shows the location of the Ythan catchment in relation to the rest of Scotland, while Figure 3 shows the soils and land cover maps as they were derived from the existing datasets.

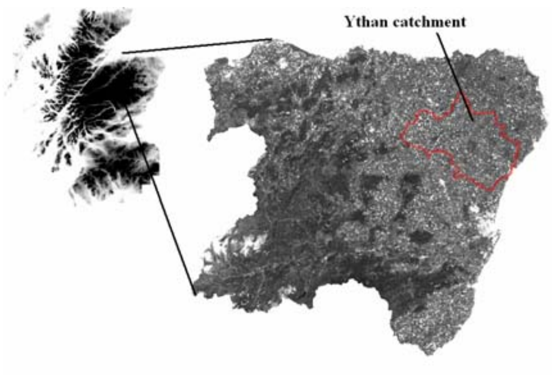


Figure 2. Location of the Ythan catchment

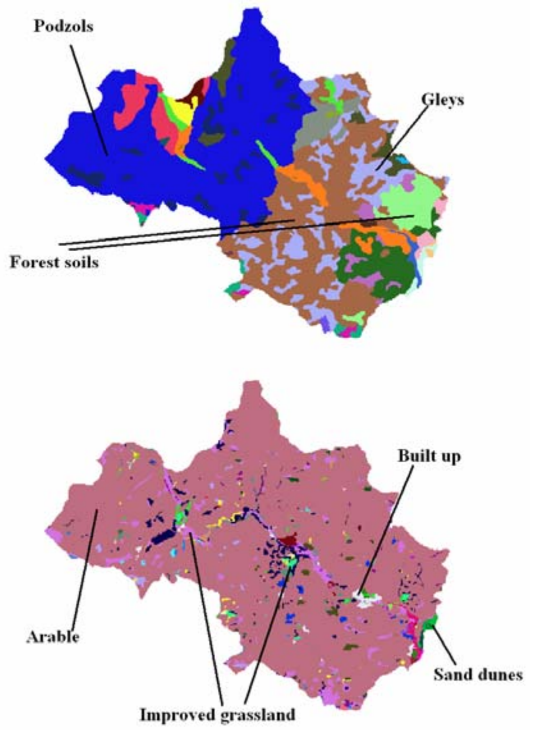
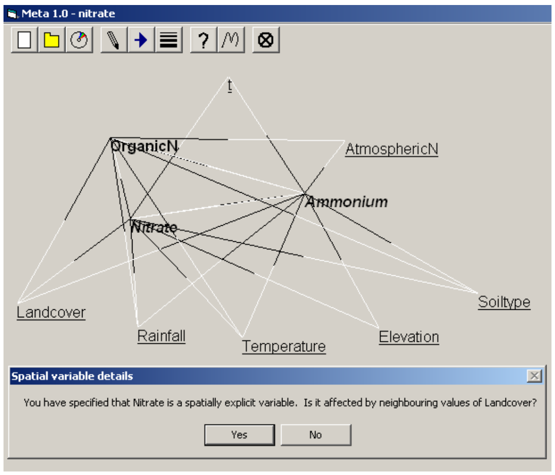


Figure 3. Soil and land cover maps of the Ythan catchment used for modelling

The development of the model was carried out by first specifying the variables to be used within the system (i.e. nitrogen pools, spatial information variables, land cover class types etc.), and then giving the relationships between and baseline data source file locations for these variables. Figure 4 demonstrates the model under development, showing the process of defining relationships that affect nitrate runoff into rivers. Meta's design is such that once variables have been defined and described, the package iteratively prompts the user for information required to run the model, continuing the process until sufficient information has been obtained. Once this point has been reached, the user can then adjust the model further, which may lead to more requests for information from Meta, or can either save the model or simply run it.

The aim of the model was to provide a simulation of the quantities of nitrogen in its various soluble forms (nitrate and nitrite) over the Ythan catchment, at a resolution of 100 metres. Inputs and outputs from each cell within the spatial model were estimated, with application of nitrogen fertiliser on arable land over the year and processes of soil solute transport, uptake by vegetation, runoff into rivers and chemical alteration simulated from available knowledge and information. The use of a separate neural network component to investigate relationships between certain variables, along with the decomposition of the trained neural networks into mathematical relationships for implementation within the model, was necessary in this case for producing relationships between water rainfall and water flow between cells, while all of the remaining relationships were available or could be approximated beforehand.



You have specified that Nitrate is a spatially explicit variable. Is it affected by neighbouring values of Landcover?

Figure 4. Screenshot of Ythan model under development

3.3. What needs to be changed

In order to develop a more realistic model of the nitrogen dynamics of an arable river catchment, two main areas of improvement have been identified. The first involves acquisition of baseline data, including more sophisticated descriptions of land use within the arable and forest classes and of descriptions of soil type. This problem does not relate directly to the Meta software, although it is important to emphasise that without good initialisation data, no model can be developed in the first place. The second second area of improvement involves the ability to implement more sophisticated processes within the model. Of specific interest to the model developed here is the change of state from one land cover class to another over time, reflecting the nutrient status of the soil at each location. In order to implement this behaviour, Meta would be able to handle state change rules, of a kind similar to cellular automaton rule sets.

3.4. Running the model

The model implementation and run showed that it is possible, assuming that all relationships are defined prior to model development, to quickly and easily produce a working model. The actual results of the model were relatively basic, and an examination of the results themselves showed that one of the main areas where improvement is required is in the storing and visualisation of model run results. The data set produced was large and difficult to understand, and required intensive effort to copy into a spreadsheet for examination. However, visual examination of the results did indicate that there were no problems with the software operation. Figure 5 gives information derived from the model results, showing nitrogen density in soil moisture after 100 1-day time steps from simulated baseline data in which the original nitrogen distribution was constant over the whole catchment.

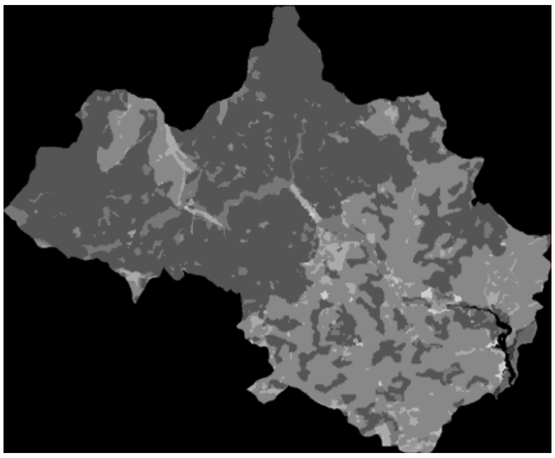


Figure 5. Model results showing soil water nitrate concentration

4. DISCUSSION

The type of software demonstrated here is useful for allowing users with no experience to develop initial models of their study system. The graphical interface is also perceived to be potentially useful as a demonstration and teaching tool, with students being able to 'play' with their models and experiment with the software's capabilities. However, it has also been demonstrated capable of allowing the user to develop a relatively sophisticated model of a spatial, complex system – a river catchment – with different types of variables and relationships implemented within the model. Intuitive user dialog algorithms allowed this model to be built with little or no previous planning or design of the model; adjustment of the model following completion, and running it in order to provide a scenario simulation of the river catchment were carried out quickly and easily. Visualisation of the model run was not possible to the level of sophistication that would be useful for understanding how the different processes within the model operated and interacted, and this area has been identified as a priority for future development.

The Meta software package was designed to be used by those with little or no programming experience, and with no desire to spend a long time learning how to use software that required in-depth training in order to develop models. This relative ease of use does come with a cost in terms of functionality; however, the range of modelling or depth within a certain topic that can be carried out in commercial packages is not available within Meta. It is extremely easy to develop relatively simple models, including spatially explicit agent-based models of complex systems, provided that expectations are not too high. As such, Meta provides a useful 'first-look' potential for modelling systems of interest, or exploring unknown relationships between components of a system.

In order to expand the capabilities of Meta version 1.0, several necessary or useful improvements to the program have been identified. These include (a) the development of a more interactive graphical user interface allowing the user to manipulate the model variables, relationships and rules directly without the need for dialog boxes, (b) an expansion of the potential relationships and variable types within models constructed using Meta 1.0, through either addition of new concepts or the generalisation of Meta's model-building routines to allow a broader range of relationship definitions on the part of the user, and (c) improvement to the graphical interface in such a way that model runs can be displayed in real-time and using a variety of different display formats, and (d) the addition of bespoke neural network modelling algorithms within the package to allow modelling directly from datasets.

5. REFERENCES

Andren, O., Lindberg, T., Paustian, K., Rosswall, T. (eds.) (1991), Ecology of Arable Land-Organisms, Carbon and Nitrogen Cycling. *Ecological Bulletins (Copenhagen), no. 40.*

Batey, T. (1982), Nitrogen cycling in upland pastures of the UK. *Philosophical Transactions of the Royal Society, London, B296, 551-556.*

Chapman, P.J., Williams, B.L., Hawkins, A. (2001), Dissolved organic nitrogen in a peaty podzol: influence of temperature and vegetation cover. In *Rees, R.M., Ball, B.C., Campbell, C.D. and Watson, C.A. (Eds.), Sustainable Management of Organic Matter, CAB International, Wallingford, UK, pp.247-255.*

Sarjala, T., Potila, H. (2005), Effect of ectomycorrhizal fungi on nitrogen mineralisation and the growth of Scots pine seedlings in natural peat. *Plant and Soil 269(1-2): 171-180.*

The Macaulay Institute for Soil Research (now the Macaulay Institute) (1982), Soil Survey of Scotland. *University Press, Aberdeen.*

The Macaulay Institute (1993). The land cover of Scotland. Final Report. Aberdeen: The Macaulay Institute.