Methods and software tools for estuary behavioural system simulation

Defra/Environment Agency Flood and Coastal defence
R&D Programme-FD 2117

Project Report 4

Jon French
Helene Burningham

DRAFT

August 2006

University College London

CONTENTS

1. Introduction

The overall aim of FD2117 'Development and Development of Systems-based Estuary Simulators' (hereafter referred to as 'EstSim') is to extend the ability to simulate the morphological response of estuaries to change. This will be achieved through the delivery of research into a systems-based behavioural modelling approach as an alternative yet complimentary methodology to more conventional modelling methodologies being researched within the other Estuaries Research Programme (ERP) Phase 2 projects (which encompasses morphological concepts, and bottom-up, top-down and hybrid modelling approaches). EstSim will also explore the simulation process in order to facilitate knowledge exchange between the systems-based tools and estuary managers.

EstSim is organised around nine scientific objectives and associated tasks:

1. System conceptualisation
2. Development of management questions [Task 2.1 to 3.4]
3. Development of behavioural statements [Task 3.1 to 3.10]
4. Mathematical formalisation [Task 4.1 to 4.8]
5. Development of system simulator [Task 5.1 to 5.6]
6. Manager system interface [Task 6.1 to 6.5]
7. Pilot testing [Task 7.1 to 7.5]
8. Dissemination [Task 8.1 to 8.6]
9. Peer review

Progress on the conceptualisation and formal definition of estuary system structure and behaviour is summarised in FD2117 Project Report 2 '*EstSim Behavioural Statements Report* 2' (EstSim Consortium, 2004). This includes a comprehensive classification of UK estuaries into seven generic estuary types. Each of these generic types has been conceptualised in terms of the major morphological and process components. The conceptualisation includes a system diagram and accompanying behavioural statements that together describe the estuary system components and their interactions. FD2117 Project Report 3 '*Mathematical Formulation of Estuary Simulator*' (Karunarathna and Reeve, 2005) outlines a mathematical formulation, based upon Boolean logic, that provides qualitative insights into the behaviour of estuary systems without the need for a physically-complete specification of the linkages between the various components of the system.

This report summarises work undertaken under Objective 5 (above). It briefly sets out the context for qualitative approaches to the understanding of estuary system behaviour before examining key issues relating to the development of a prototype estuary system simulation software tool:

1. the computational basis of qualitative behavioural system models in general, and the Boolean network approach in particular;

2. alternative software platforms for behavioural estuary system modelling;

3. alternative architectures for the development of a working prototype simulator that can contribute to the broader set of estuary modelling tools developed under the ERP.

4. results obtained using a prototype MATLAB-based simulator applied to the generic UK estuary types identified in FD2117.

2. Qualitative modelling of estuary system behaviour

As is the case generally in geomorphological systems, estuarine and coastal morphodynamic behaviour is complex because of the feedbacks between morphology and sediment transport. An important consequence of these feedbacks is that the operation of a system at any time is influenced by previous states (geomorphologists refer to this as *state dependence*, or *inheritance*). Additional complexity arises from the interplay between self-regulation (or *equilibrium tendency*) and self-forcing (which leads to *thresholds* and *complex response*), and the non-linear nature of many of the functional linkages between system components (see, for example, Wright and Thom, 1977; Cowell and Thom, 1994). Equilibrium represents a morphodynamic state that is stable for a given set of environmental boundary conditions. This stability can take the form of a steady state or oscillation about a long-term average condition. Many geomorphological systems can also potentially exhibit a chaotic equilibrium state (Phillips, 1992).

The identification of equilibrium conditions and the manner in which estuary morphology adjusts towards a new equilibrium state following a change or perturbation in system operation are of great interest to estuary managers and users. In engineering, much use has been made of so-called *regime models*, which determine directly the stable morphology that arises from a balance between sediment transporting forces. These states can be determined analytically or numerically (in the case of some shoreline models) or empirically (e.g. tidal inlet scaling relationships of the kind developed by O'Brien (1969) and re-evaluated by Townend (2005)). These are effectively top-down models in the sense used in the ERP (EMPHASYS, 2000). Alternatively, the time-evolution of coastal or estuary morphology can be modelled explicitly. This is typically attempted through *process-based morphodynamic models* (bottom-up models in the sense of the ERP). These contain detailed mathematical formulations for hydrodynamics and sediment dynamics and are computationally intensive. Accordingly, they are usually executed in some form of time-stepping framework (e.g. de Vriend *et al.*, 1993) in which the time-step for morphological change is much greater than that for the simulation of physical processes. Process-based morphodynamic models provide considerable insight into short-term system behaviour, but long-term evolution towards equilibrium is often incorrectly reproduced (see, for example, Hibma *et al.*, 2004).

None of the above approaches are suited to the modelling of complex systems for which the functional relationships between morphological components and driving processes cannot all be mathematically specified in a way that is physically realistic. This is certainly the case for whole estuary systems, where changing environmental forcing factors (e.g. sea-level and sediment supply) and constraints (e.g. structures) may prevent evolution to equilibrium morphology, and for which quantitative functional representations are not available for many of the linkages and feedbacks between system components.

A qualitative approach addresses many of these problems and provides an alternative basis for modelling the behaviour of estuary (and coastal) systems, as abstracted at the scales of most relevance to the tackling of estuary management questions (i.e. the 'engineering scale' of Cowell and Thom (1994) and the 'meso-scale' of Townend (2003)). A formal systems-based approach to the conceptualisation of estuarine and coastal systems, supported by the technical vocabulary of general systems theory (von Bertalanffy, 1968; Bennett and Chorley, 1978), has been advocated by Townend (2003). The potential of qualitative modelling as a means of providing indicative rather than strictly quantitative insights into the behaviour of systems specified in this manner has been highlighted by Capobianco *et al.* (1999).

Capobianco *et al.* (1999) identify seven stages associated with the development of a qualitative model of system behaviour (Figure 1). These include the identification of qualitative variables and the causal linkages between them, which typically involves the construction of system diagrams of some form. Modelling of system behaviour then requires the definition of a mathematical quantity space to represent interaction between the state variables. Figure 2 shows a generic tidal inlet system subject to a rise in sea level, wherein the quantity space is defined in terms of a signed graph, with positive or negative effects connecting the main system variables. Transfer rules and knowledge formalisation are then needed to convert either quantitative understanding (which might take the form of empirical scaling relationships) or linguistic understanding (e.g. a statement describing the behaviour of a sub-component) of system linkages into a set of cause-effect relationships. Qualitative calculus refers to the employment of mathematical analyses that treat the system as an interconnected network (or graph) in order to achieve insights into the direction of change in the state of any or all of the system variables in response to a change (such as external forcing or an imposed alteration to system structure).

As highlighted in the ERP Phase 2 Research Plan (French *et al.*, 2001), top-down qualitative modelling of this kind may provide extremely useful knowledge of estuary behaviour at the scales most relevant to managers and users. New emergent properties of system behaviour may be revealed, including the existence of unexpected sensitivities to change and/or constraints upon the evolution towards morphological equilibrium. Qualitative judgements concerning estuary behaviour also provide a basis for evaluating the output from quantitative models (i.e. in terms of the direction of predicted change rather than its magnitude). Such insights can be crucial in impact and vulnerability assessment.

3. Computational basis of qualitative behavioural system models

3.1 General issues

Qualitative modelling of system behaviour requires protocols and/or tools for system conceptualisation, mathematical representation, analysis and visualisation. The computational requirements are rather different from those of quantitative process-based models and the analyses performed are typically much less demanding, such that meaningful simulations can be implemented and executed with quite limited software and hardware resources.

In terms of their implementation in software, algorithms for mathematical representation and analysis are less computationally challenging than those used in quantitative process-based models. Most forms of qualitative calculus analyse systems as interconnected networks. The vertices of these networks are the system components and the vertices represent functional linkages between them (Figure 3). Examples abound in biomedicine, where the subject of study includes gene expression networks, biochemical circuits in cells, neural networks (e.g. Glass, 1975; Thomas, 1979; Fox and Hill, 2001), and the World Wide Web has even been analysed in this manner (Barabási *et al.*, 1999). There have been fewer applications to the study of environmental systems, with most of these being in the area of climate modelling (for example, Nicolis, 1982; Saunders and Ghil, 2001).

1. *Identification of qualitative variables* and the likely relationships between them
2. *Conceptual representation* and definition of causal inference network
3. *Definition of a quantity space* – a finite set of symbols to define values of state variables
4. *Establish transfer rules* that can be used to develop qualitative relationships between variables
5. *Knowledge formalisation*, in which transfer rules are used to define cause-effect relationships
6. *Qualitative translation of measurements*, where variables are assigned values from the quantity space
7. *Application of qualitative calculus* to analyse characteristics of model

Figure 1: Construction of a qualitative model (adapted from Capobianco *et al.*, 1999).

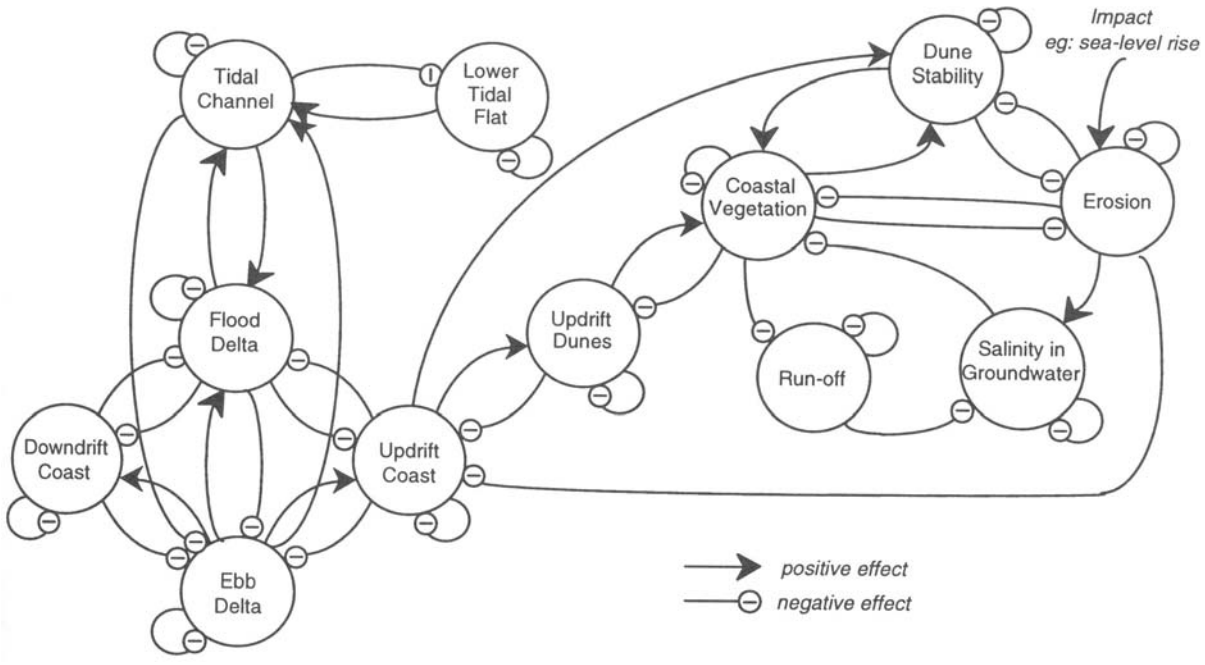


Figure 2: Example of a qualitative model of the impact of sea-level rise on a hypothetical tidal inlet (modified from Capobianco *et al.*, 1999). This is probably the simplest form of model, in which system inter-linkages are represented merely as directed influences (i.e. in the form of a 'signed graph').

The interconnections in a qualitative model are represented by fairly simple mathematical functions. Synchronous models start with an initial condition and simultaneously update the values of all state variables (components) such that the system evolves over successive steps. Depending on the mathematical formulation of the interconnections, these steps can either be thought of merely as sequential states that do not correspond directly with time, or as discrete time steps. In more sophisticated formulations, updating is asynchronous thereby allowing the time-scales of system behaviour to be investigated directly. Even for large systems, the analysis of individual scenarios (i.e. evolutionary behaviour from a given initial condition) requires few computational resources. However, we are often interested in exploring a system's parameter or state space (for example, to determine the proportion of the potential system states that are likely to actually occur in nature, or to search for potentially 'disordered' behaviour). As the number of potential system states increases exponentially with the number of components in the system, such evaluation of the entire parameter space becomes impractical for systems of even moderate complexity. For large systems (hundreds or thousands of components), understanding of behavioural properties must come from sampling.

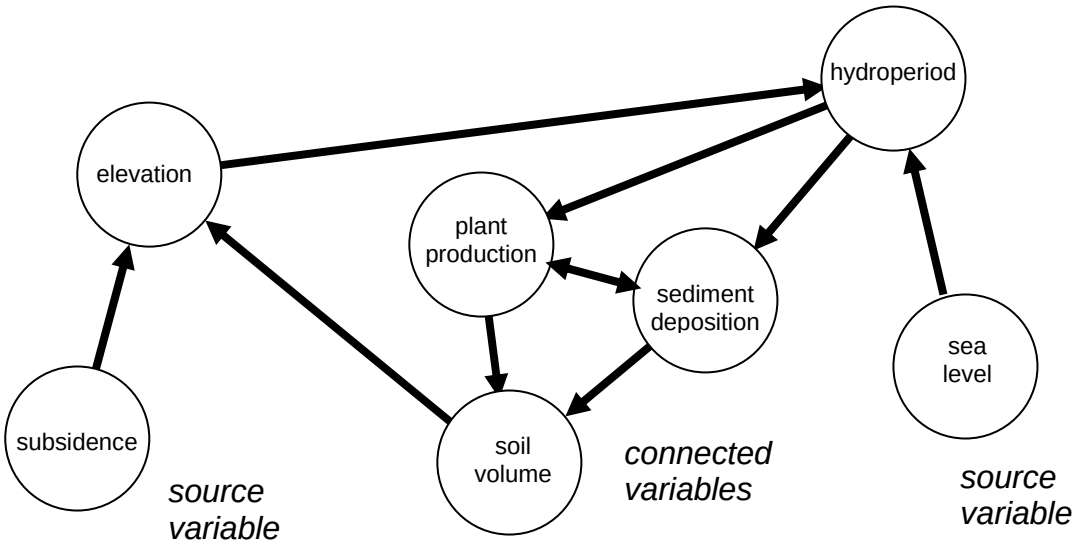


Figure 3: Simple network representation of saltmarsh system that includes source variables (that influence the network dynamics but are not influenced by the network) and connected variables (that both influence and are influenced by network dynamics). Sink variables and unconnected variables can also be defined, although these are less useful for the representation of environmental systems.

3.2 Boolean network formulation

Of the various qualitative simulation approaches available, the Boolean network formulation is particularly easy to implement. As outlined in FD2117 Project Report 3 (Karunarathna and Reeve, 2005), a system is defined in terms of a network of N components, which are represented by Boolean state variables. These are assigned a binary value (i.e. either 0 or 1). The value of each state variable is determined by a Boolean function that is coded using logical expressions (based on logical NOT, AND, OR operators) that incorporate the combined influence of other state variables to which they are connected. System behaviour is simulated by specifying a set of initial conditions (i.e. values of all state variables) and evaluating the set of Boolean functions to arrive at a modified set of state variables. This is repeated until the system reaches an equilibrium condition.

Two kinds of equilibrium are routinely encountered. A steady state exists when all the state variables take the same values as their corresponding Boolean functions, and no further change is possible. An oscillatory state is possible, when the system falls into a cyclic sequence between two end states (possibly involving additional intermediate states). These states can be thought of as attractors within the state space defined by all possible system states. Studies by theoretical biologists of large Boolean networks ($N > 100$) indicate the possibility of a third 'disordered' end state, in which successive states appear non-repeating up to some arbitrary number of steps (Fox and Hill (2001) use a cut off of 1500 steps to define the boundary between periodic and disordered states). Estuary systems investigated at a meso-scale are readily conceptualised in terms of a much smaller number of state variables ($N < 20$). System behaviour should thus be ordered, even though both steady state and cyclical end points are possible.

Simulation based on a Boolean network approach can proceed in either of two ways. In the first, we examine the range of behaviour that is encountered across the system state variable space. Since the number of possible states is 2^N, exploration of the entire state space is only possible for simple (i.e. small N) systems. For more complex (larger N) systems, some form of statistical sampling of the range of possible initial conditions is required. Studies of this kind can yield useful statistics concerning aspects of system behaviour such as:

- the number of attractors (steady state and oscillatory equilibrium states);
- the sequence lengths of oscillatory states;
- the distribution of evolutionary sequence lengths (including maximum path length);
- the proportion of potential states that occur as viable intermediate states.

Theoretical studies of biological and other networks show that number of attractors (equilibrium states) depends on N, and also on the number of input connections between state variables, K. For fixed $K = 2$ systems, Kaufman (1996) estimates the number of attractors to be approximately $\sqrt{N}$, with the length of oscillatory attractors also given by $\sqrt{N}$. As systems become large, the attractors thus account for a tiny fraction of all possible system states. Other studies (e.g. Bilke and Sjunnesson, 2002) show that higher K-connectivity and variable K-connectivity (as is the case in network representations of geomorphological systems) are associated with increased complexity (as measured by the number of attractors and sequence lengths). Complexity also increases with the number of source variables (that is state variables or system components that influence the network, but are not influenced by it). In the case of geomorphological systems, such additional complexity might be expected to result from multiple external forcing factors (e.g. sea-level, storminess, sediment supply).

The second kind of simulation involves a more intensive analysis of the evolution of the system from specific initial conditions. Analysis at this level involves a more detailed interpretation of the evolution of system states against scientific understanding, as encapsulated in the behavioural statements associated with the components of the estuary system. Validation of a model (in terms of the set of state variables and the formulation of the Boolean functions) can be carried out in the case of estuaries for which sequences of historic behaviour are documented, and 'what-if?' scenario modelling can be undertaken in response to particular management questions.

It is envisaged that a prototype estuary simulation software tool based upon this approach will be primarily used for the second kind of analysis. Estuary users should be able to select one of the generic UK estuary system types and evaluate its behaviour under specific scenarios that reflect management questions of interest. However, as part of the proof of concept, investigation of the state variable space for different kinds of estuary is also of interest.

4. Software platforms for behavioural estuary system modelling

There are various software options for qualitative modelling of system behaviour. In selecting the most appropriate of these options, an overriding criterion should be the purpose for which the modelling is being undertaken and the intended user. Once this is established, the pros and cons of particular software packages come into play. In the context of FD2117, behavioural system modelling software is needed to support the following activities (and users):

- evaluation of the Boolean network approach formulated for FD2117 (EstSim consortium)
- development of architecture for a Boolean network-based simulation tool (EstSim consortium; subsequent developers of tools based on this approach)
- pilot testing of prototype simulation tool (EstSim consortium; consultants and researchers)
- dissemination of research, including prototype/demonstration tool (users and consultants)

Computation of evolutionary behaviour for a simple Boolean network can, in principle, be undertaken using readily available spreadsheet software (e.g. Microsoft *Excel* or OpenOffice *CALC*). However, for research into the properties of a range of estuarine systems, or for the implementation of a user-friendly estuary management tool, spreadsheet-based software would represent a clumsy and inefficient solution. More suitable platforms include:

- Specialised system simulation packages (e.g. *Simile, VisSim, Stella, ModelMaker, VenSim, Extend*).
- Numerical analysis and visualisation packages (e.g. *Matlab, IDL, Mathematica, MathCad, Octave*).
- High-level programming languages (e.g. C, Fortran, BASIC, Java), including some that are specifically intended for system modelling applications (e.g. *Simula, PCRASTER, VisSim*).

With the objectives and target uses of EstSim in mind, relevant criteria for the selection of an appropriate software platform for the development of a Boolean network-based estuary simulator include:

- Sufficient programming capability (data types, logical and other operators, instructions and control flow, and file handling) to permit research-level use in support of proof of concept testing
- The ability to integrate underlying model code (i.e. Boolean functions and control flow logic) with textural and/or graphic representations of system structure (e.g. behavioural statements, system diagrams).
- Visualisation capabilities.
- User interface capabilities.

- Delivery options (e.g. the ability to produce a distributable standalone software tool; support for server-based WWW delivery, or the ability to distribute model codes as 'plug-in' scripts for use with either proprietary or open-source application software).

- Support for open source model code and/or application software.

- Level of required expertise (ease of use and degree of training required for users).

- Cost and licensing arrangements (especially for users outside academia).

The following sections evaluate the capabilities of leading proprietary and open-source software against these criteria. The review focuses on two classes of software. First, packages that allow the user to construct graphical representations of the system of interest before interactively specifying the associated functional linkages and dependencies. Second, software that requires the user to implement the underlying model algorithms in some form of computer code. High-level computer programming languages are not considered further here, since these are effectively capable of implementing any software solution (though often at considerable cost in terms of development time).

4.1 Specialised system simulation packages

There is now a large selection of specialised packages for systems-based simulation modelling. Many of the most sophisticated (and expensive) packages have been developed for commercial applications, notably the simulation of business organisations (up to the scale of national public sector bodies and major multi-national corporations) and industrial production processes (including real-time process control applications). Relatively few packages are specifically intended for application to the understanding of environmental system behaviour.

One of the pioneers in the application of systems thinking to the modelling of environmental systems was *Stella*, released by High Performance Systems, Inc (now iees systems, inc) in 1987. This object-oriented software allows the user to define the structure of the system of interest with the aid of graphical 'drag and drop' tools and pre-defined component types, and then to specify functional linkages interactively using libraries of mathematical operators and integration algorithms. Interactive dialogue windows also facilitate specification of boundary and initial conditions (which may be read from external data files) as well as run control information (e.g. time step and simulation duration). Models developed in this way are saved in a high-level 'pseudo-code' (sometimes referred to as a Modular Modelling Language, or MML), which is then interpreted into lower-level computer code when the model is run. Although this approach is computationally inefficient for the implementation of complex numerical algorithms of the kind deployed in bottom-up hydrodynamic and sediment transport models, it provides a highly intuitive platform for problem solving within a systems framework.

Applications of *Stella* have included the prototyping of coastal wetland models that can subsequently be re-coded into a high-level programming language and embedded within large scale spatial landscape simulations (Sklar *et al.*, 1994). Software of this kind allows rapid conversion of concepts to logical and mathematical expressions, while preserving a graphical representation of the linkages among variables. Combining a conceptual representation of system structure with a pseudo-code formulation of the associated mathematical algorithms in a runtime environment results in models that are particularly easy to understand and use by non-specialists.

A variety of other products now offer similar functionality and Appendix 1 summarises the main features of those that are most obviously suited to environmental modelling applications. Products vary widely in capability, ease of use and cost. The cheapest and easiest to use is probably *ModelMaker*, which offers fairly rudimentary model building and simulation capabilities, with only limited communication with other applications. At the other extreme, are heavyweight business simulation tools, such as *PowerSim* and *VisSim*, which provide sophistication modelling and application development features. *Extend* is also more obviously suited for business applications, though Odum and Odum (2000) demonstrate its potential as a tool for environmental system simulation. Given their commercial applications, such products are expensive, particularly for non-academic users. The main competitors to *Stella* in the mid-market range are probably *Simile*, which offers similar functionality at lower cost and *Vensim*, which is similar in cost but more targeted at business applications.

Comparative evaluation of these various packages in terms of their suitability for behavioural modelling of estuarine or coastal systems should clearly take account not just of their technical capabilities and cost, but also factors such as ease of use, vendor support (documentation, technical support), and user-community (including existence of established online resources and user forums). Also important, in the case of EstSim, is choice of architecture for any software tool that is developed and disseminated amongst the user community. However, a preliminary analysis highlights *Stella* and *Simile* as mid-range products that offer a good combination of functionality, ease of use, widespread take-up within the academic community and cost. *Stella* appears to be more widely used as an educational tool, and is strongly featured in textbooks on system modelling (e.g. Ford, 1999; Odum and Odum, 2000). However, *Simile* scores highly on its documentation, interoperability with other application software and suitability for environmental system simulation (see, for example, Mulligan and Wainwright, 2004), and is significantly less expensive. In addition, the formal logic used for model conceptualisation in *Simile* is probably more intuitive than that employed by *Stella* for users trained in geomorphology. Heavyweights such as *PowerSim* and *VisSim* probably represent overkill for the relatively simple modelling tasks envisaged here and would be expensive to roll out amongst a large and distributed user community. However, they might be a viable option for the implementation of web-based simulation tools utilising server-based computation.

4.2 Numerical analysis and visualisation packages

A number of sophisticated packages are available that integrate the power of a high-level programming language (such as C or Fortran) with easy to use provide tools to support algorithm development, data handling, visualisation and even the creation of standalone application software with full graphical user interface capabilities. Appendix 2 summarises the products reviewed here. Widely used commercial packages include *Mathematica*, *Matlab*, *IDL* and *Mathcad*. Open-source offerings include *Octave*, which provides a library of functions intended to offer comparable function to *Matlab*. All the major commercial softwares offer:

- A *development environment* - including command line interface, script editor and debugger, and browsers for viewing help, workspace variables and files.

- A *library of mathematical functions* – including computational algorithms ranging from elementary functions, solvers for ordinary and partial differential equations.

- A *programming language* - possibly supporting matrix/vector operations, with control flow statements, functions, data structures and data file handling. Support for interactive

'programming' of simple tasks as well, and well as the development of complex applications.

- A *graphics system* – including high-level functions for data visualization, image processing, animation, and presentation graphics; and low-level functions for custom graphics the design of graphical user interfaces for applications.

- An *Application Program Interface* - to permit interfacing with external Fortran or C programs.

- Optional *extensions or toolboxes* – to provide additional functionality (and function libraries) to handle specialised areas (e.g. image processing, signal processing, neural networks, or even system simulation (in the case of the *Matlab-Simulink* module)

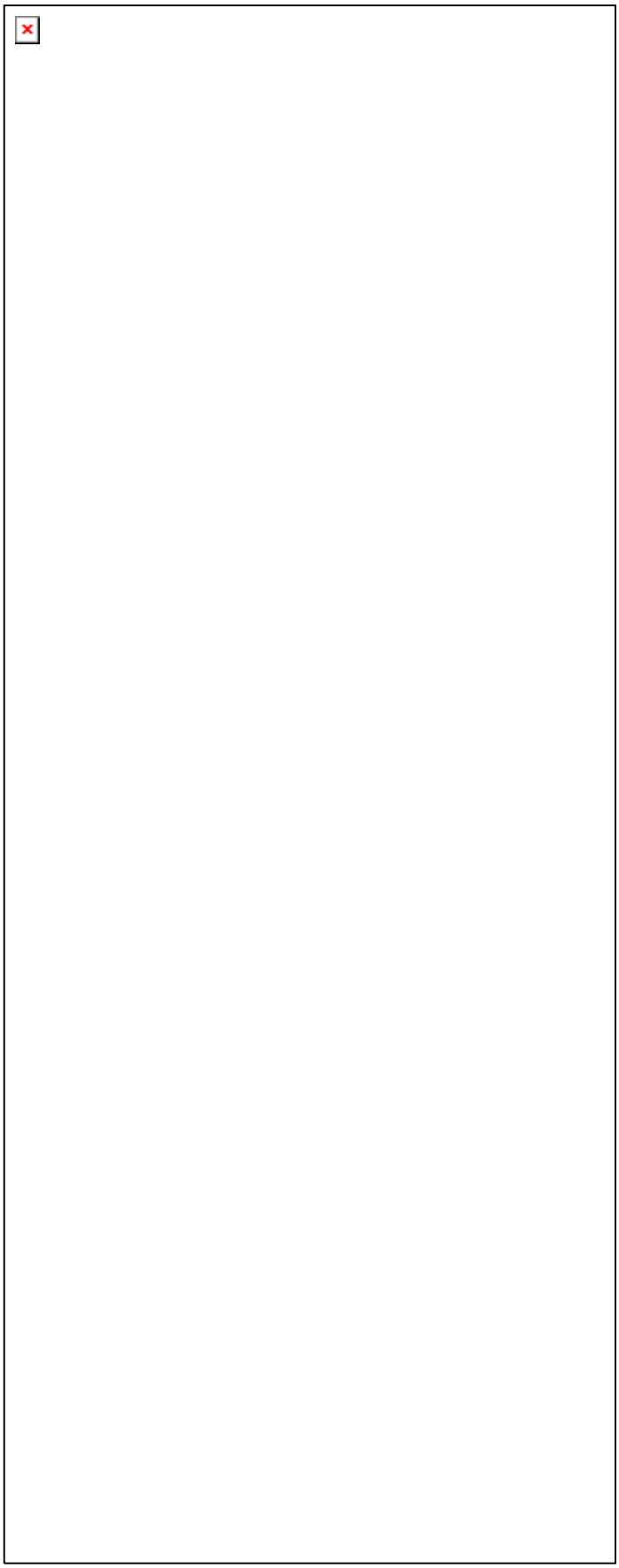
- Optional *compilers* – to permit the development of standalone applications that can be distributed and executed independently of the originating package. Compilation is also useful for accelerating the performance of frequently-used scripts (which run slowly in interpretive mode).

These packages are all suitable for fairly large and complex modelling tasks and they are more than capable of handling any computational aspect of behavioural system modelling based on a Boolean network approach.

The commercial packages are well supported and have established user-communities that contribute algorithms, scripts and even complete toolboxes to online archives. *Matlab* and *IDL* are probably most useful for reasonably efficient numerical computation of fairly large problems, whilst *Mathcad* and *Mathematica* are particularly strong on symbolic mathematics and equation solving. *Octave* is the only open-source product that falls into this category. Despite its capabilities, this must be regarded as a research tool rather than a mainstream software development product: the technical expertise required for its installation and use and its lack of application development features effectively preclude its selection as a development platform for an EstSim tool.

All are extremely powerful tools in experienced hands, but the learning curve is very high and these are not easy to use packages for the casual or novice user. The commercial packages are also expensive for non-academic users. Software of this kind can be classed as 'open-ended', since scripts are executed (and can be freely distributed and modified) as ASCII-text files (although the original code can be 'protected' if desired by partially compiling into a proprietary pseudo-code). Another feature of these packages is that most are truly multi-platform, with Windows, UNIX, Linux and MacOS-X systems all supported).

Table 1 presents a subjective evaluation of the various software products reviewed here against criteria relevant to qualitative modelling of estuary system behaviour.



5. Alternative architectures for an EstSim software tool

Implementation of a fully featured web-interfaced estuary behavioural system simulation package is beyond the immediate scope of FD2117. However, it should be possible to devise a fairly simple software product that provides essential R&D experience (including proof of concept testing and evaluation of system conceptualisation undertaken in this project), as well as an easy to use tool for dissemination of the behavioural systems concept and for the rationalisation of management questions that can be addressed through a systems approach.

Various possible architectures for an EstSim tool can be envisaged based around combinations of functionality and delivery. In terms of *functionality*, simulation of estuary system behaviour might take the form of interactive selection from a fixed set of scenarios interfaced to pre-computed model outcomes. The ability for the user to define custom scenarios and/or system structures would require on-demand computation and a more fully featured software tool. The highest level of functionality would also allow 'research-level' uses, such as exploration of a system's state variable space to analyse more generic aspects of its behaviour.

There is, likewise, a range of options for *delivery*. These include distribution of a standalone software application (either on CD or via the World Wide Web); implementation of a web-based decision support system; and implementation of a more sophisticated web-based decision support and analysis tool interfaced to server-side computation facilities to support a full spectrum of interactive analyses into system behaviour.

Variations on these themes are also possible: a web-based decision support system could easily be supplemented by downloadable tools and/or customisable models (either standalone products or compatible with one of the leading proprietary system simulation packages). Some alternative architectures that would meet the immediate requirements of FD2117 are summarised below.

1. *Web-based estuary simulator and prototype decision support system*
A model for this is ABPmer's online Estuary Guide (www.estuary-guide.net). An EstSim tool based on this architecture might comprise:

- a browser-based interface to the EstSim classification of UK estuaries, their generic system diagrams, and behavioural statements for their associated sub-systems.
- a database of pre-computed system evolution trajectories with accompanying textual descriptions, linked to a menu of estuary types and most likely management questions.
- a small database of worked examples to demonstrate the robustness of the underlying behavioural system approach when applied to test case estuaries for which historical information is available.

This essentially constitutes a web-linked interactive database in that it has no computational capability that would support editable system diagrams or user-specified scenarios. However, it could still serve as a valuable demonstration and proof of concept tool.

2. *A web-based estuary simulator with server-side computational capabilities*
This approach would offer all of the above, with the added functionality of:

- server-based computation of user-specific scenarios, possibly with support for customisation of estuary system diagrams to allow investigation of actual management questions for particular estuaries.

This would produce a powerful web-based application that would be a valuable addition to the existing suite of estuary modelling and management tools being developed under ERP. The simplest and least expensive approach would use a proprietary scripting or high-level programming language to develop a custom processing engine. More complex and expensive implementations might use a heavyweight simulation package such as *PowerSim* or a *webMathematica* server. In all these cases, however, the development effort required to produce a polished product would be considerable, such that this option does not seem feasible for EstSim.

3. *Standalone estuary simulator*
An EstSim tool of this kind might comprise:

- a pre-compiled simulator developed using either a specialist system simulation package (e.g. *Stella*) or a compiled GUI-tool developed using a numerical computation package (e.g. *MATLAB*).
- the simulator would include pre-defined system diagrams and a set of pre-defined scenarios, but would also allow the user to specify custom system diagrams or scenarios through external (open-format) run control or steering files.
- a more sophisticated implementation might include simple graphical tools or dialogue boxes for customising system diagrams and scenarios, although this would involve considerable software development effort that is probably beyond the scope of this project.
- a supporting web site from which the tool could be downloaded as freeware, together with model files for all UK generic estuary types. Parallel distribution on CD would also be possible.

This might produce a very useable product, with a simple GUI-based tool developed in (say) *MATLAB* probably offering greater ease of use than a pre-packaged *Stella*-type model (which would require users to learn how to use what is still quite a complex piece of application software). In the case of a tool built using *MATLAB* (or similar), it would be possible to maintain some of the research-level functionality for R&D into generic properties of system behaviour: this could be de-activated in a version distributed amongst a broader user community.

4. *A semi-standalone estuary simulator*
A variation on the above architecture might involve:

- a set of pre-defined model files (system diagrams, datasets and run control files) for application with a freeware or low-cost run time version of a proprietary system simulator (e.g. *Simile, Stella*)
- a supporting web site from which the model files could be downloaded as freeware, with links to freeware versions of required simulation package. Parallel distribution on CD might be possible although there are licensing issues associated with the re-distribution of third party products, even where these are available at no cost from the vendor.

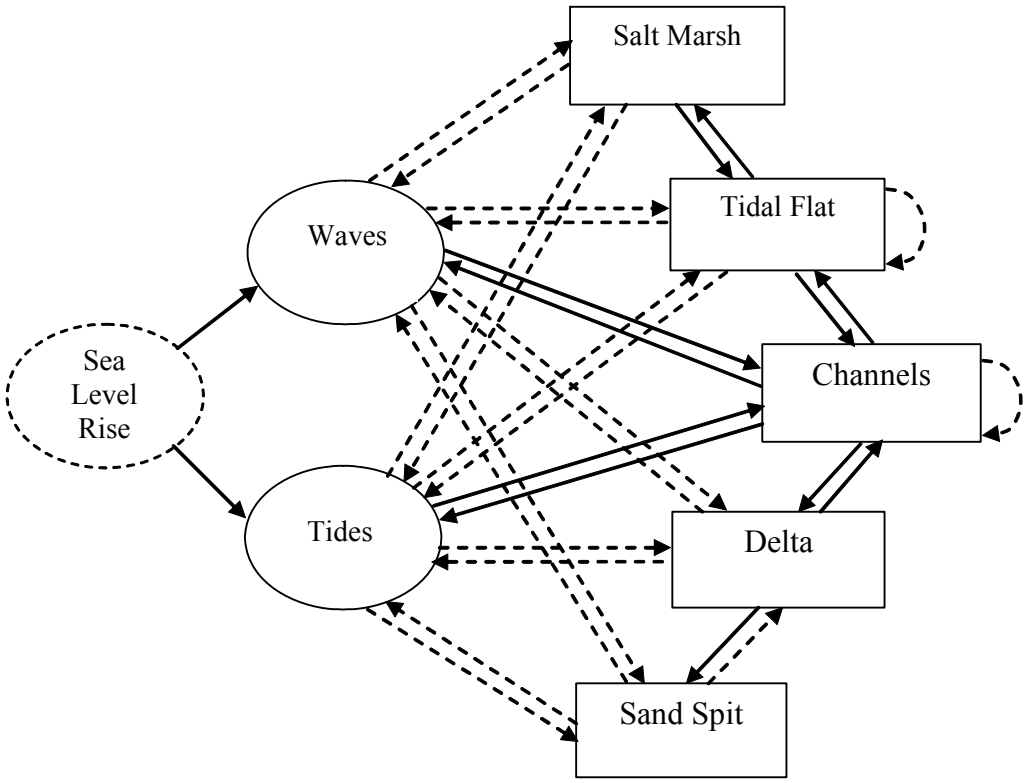
This is the least satisfactory of the above options, in that it ties the simulator to a specific proprietary package on which users must receive training.

It should be possible to implement a simulator based upon approach 1 for the purposes of FD2117. However, a more fruitful approach would probably be to combine elements of approaches 1 and 3. In the longer-term, and given sufficient resources, it might be desirable to develop a more sophisticated web-enabled product based on approach 2.

6. Boolean network based estuary behavioural system modelling

6.1 System diagram definition and specification of Boolean functions

Karunarathna and Reeve (2005) presented a preliminary Boolean network analysis for a simple tidal inlet comprised of seven components. Five morphological components were included (saltmarsh, tidal flat, channels, tidal delta and sand spit) and two external forcing factors (waves and tides). Their scheme is reproduced here in Figure 4. Whilst this model is extremely useful as a proof of concept exercise, the generic estuary behavioural types presented in FD2117 Project Report 2 '*EstSim Behavioural Statements Report* 2' (EstSim Consortium, 2004) are conceptualised in rather more detail (the generic tidal inlet, for example, comprises ten morphological components) and there are some differences in the functional logic (i.e. in the interconnections between components).



Boolean and variables and functions for the above system

Waves: W = ~sm | ~tf | cc | ~dd | ~ss;
Tides: T = ~sm | ~tf | cc | ~dd | ~ss;
Saltmarsh SM = (~w & t & tf);
Tidal flat TF = ((sm | cc) & t) | (~tf & ~w);
Channels CC = (w | t) & (tf | dd) | ~cc;
Tidal delta DD = (~w | ~t) & ~ss | (t & cc);
Sand spit SS = (~w | ~t) & dd;

Figure 4: Proof of concept Boolean network model for simplified generic tidal inlet (taken from Karunarathna and Reeve (2005)). The logical operators used are AND (&) and NOT (~), and OR (|).

At the outset, it is also clear that this simple implementation of a Boolean network approach has some important limitations with regard to its ability to represent key aspects of estuary system behaviour. These include:

1. Terminology of abstraction. For process variables, it is not always clear which metric is most relevant. In the case of tides, for example, the presence or absence of tidal action is not very meaningful. Tidal forcing, as an external factor, might be conceptualised in terms of high and low tidal ranges (effectively an energy measure). Estuary morphology might then exert some form of feedback, through dissipation or amplification of the tide, although capturing this would require additional variables. Alternatively, various aspects of estuary morphology have been shown to scale with tidal prism. Prism would seem to be a more appropriate measure, since it incorporates an implicit feedback between changes in estuary form and tidal flow.

2. *Threshold assumption of the Boolean approach*. The assumption that interactions between variables take the form of a threshold-type function (i.e. 'on / off' or 'presence / absence') is not always easy to justify. Sometimes a simple binary representation is insufficient to capture the required range of behaviour or states. In particular, increases in external forcing (e.g. the effect of a rise in sea-level, through an increase in wave and/or tidal action) are difficult to represent using on/off (or presence/absence) logic, certainly when the level of system abstraction is high. Equally, it is impossible to create additional saltmarsh (for example by setting back reclaimed tidal floodplain) in a system that has already evolved to a state in which saltmarsh is extensive.

3. *Aspatial representation of spatially-distributed phenomena*. The systems approach as used in EstSim generalises key aspects of estuary behaviour in an aspatial manner, yet many estuaries vary markedly in the behaviour of their outer and inner zones. For example, the Ribble estuary in north-west England comprises an outer estuary dominated by sand flats and an inner estuary in which mudflat and saltmarsh are more extensive. Each zone has experienced distinct but rather different historic changes and these are not easily captured in an aspatial model. This is not a limitation of a Boolean network model per se, but it does suggest the need to handle the simulation of spatially distributed behaviour, and that this might be achieved by the specification of separate, linked, sub-systems.

6.2 Further development of Boolean network model

Further development of the Boolean network approach formulated by Karunarathna and Reeve (2005) has been undertaken using a prototype simulator. *MATLAB* was selected as our preferred development platform on account of its powerful programming language and strong numerical computation and visualisation capabilities, as well as the support that it provides for the subsequent development of compiled standalone tools.

The estuary system conceptualisation has been extended to incorporate a basic spatial division into an *outer estuary* sub-system (which interacts with open coast and potentially contains beach, spit, dune, and tidal delta or linear bank features, as well as sand or mud flat and saltmarsh) and an *inner estuary* sub-system (which is dominated by an assemblage of sub-tidal channel and intertidal flat and saltmarsh features). This scheme resembles the estuary facies model of Dalrymple *et al.* (1992) in that outer and inner estuary zones are likely to be dominated by wave and tidal processes respectively.

Once the level of abstraction is clearly established, it should be fairly straightforward to reach a consensus concerning the set of variables to be included. For convenience, we distinguish between three distinct types of system component:

1: External (imposed) forcing factors
- ocean waves
- longshore wave power
- sediment supply (marine sand and marine mud)
- wind regime (favourable for dune formation)
- sea-level rise

All of the above are imposed as part of the initial conditions and are not changed by the evolutionary behaviour of the estuary. They can be altered to accommodate a change in boundary conditions, such as updrift coastal protection blocking the supply of marine sand, and specific external factors can be defined as require in order to handle anthropogenic influences such as dredging or construction of a tidal barrage.

2: Process state variables
These represent processes (*sensu lato*) that may be changed by the evolution of the morphology. State variables are presently defined to represent estuary waves and tidal prism:
- outer estuary waves (influenced by factors such as tidal prism and the protection afforded by a spit or tidal delta)
- inner estuary waves (influenced by propagation of waves from outer estuary as well as inner estuary prism and/or presence of saltmarsh)
- outer estuary tidal prism (determined by infilling of the accommodation space by tidal flat/saltmarsh, as well as by inner estuary tidal prism)
- inner estuary tidal prism (determined by infilling of the accommodation space by tidal flat and saltmarsh and by infilling of channels, and also influenced by extent of reclaimed floodplain)
- residual_transport for bedload (~sand), defined according to a simple approximation of intertidal area and depth and HW and LW (an attempt to implement a Dronkers (1986) γ-type velocity asymmetry)
- residual transport for suspended load (~mud), defined according to the intertidal morphology such that accumulation is favoured by higher intertidal (and, especially, saltmarsh as an efficient sediment sink)

3: Morphological components
A basic set of components includes:
- beach, spit and dune units (outer estuary)
- various forms of tidal delta (outer estuary)
- tidal sand and mud flat (lower and upper units, which can co-exist)
- lower and upper saltmarsh units (lower and upper units, which can co-exist)
- a rock platform that can be revealed by stripping of intertidal sedimentary cover if wave energy is high and the sediment supply negligible
- reclaimed floodplain that can contribute additional prism if setback through managed realignment
- subtidal deposition (infilling of the estuary channel) when sediment gradient is positive and in the absence of dredging

Intertidal depositional features are represented as vertically-stacked sedimentary units that are not mutually exclusive but which can co-exist (for example, mud flat and low or high saltmarsh). This

improves the ability to resolve system responses to change whilst retaining a Boolean representation of individual components. Channels are not represented as a discrete component but are incorporated implicitly as part of a subtidal morphology that can undergo deposition and erosion depending on the availability of sediment and upon changes in prism. The basic stratigraphic framework for sand and mud dominated outer estuary sub-systems is summarised schematically in Figures 5 and 6.

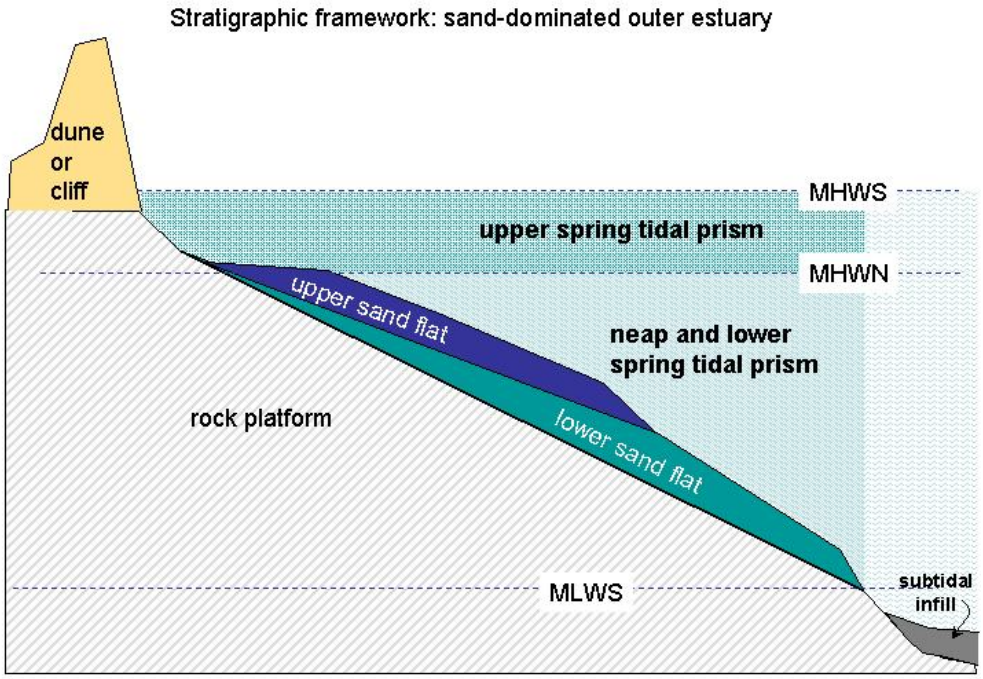


Figure 5: Schematic illustration of stratigraphic framework for sand-dominated outer estuary.

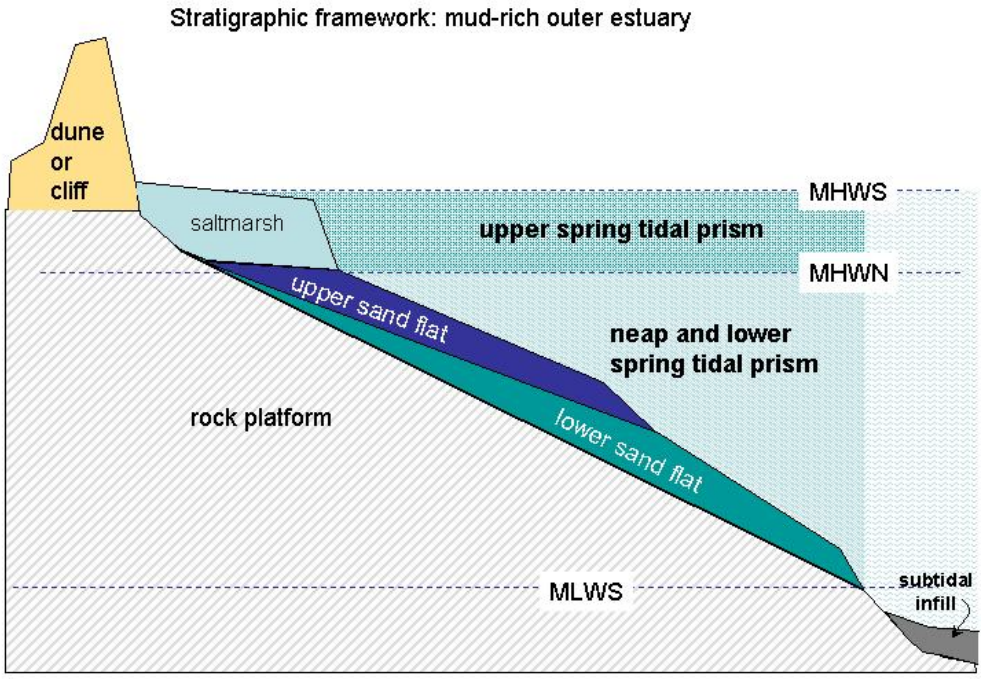


Figure 6: Schematic illustration of stratigraphic framework for mud-rich outer estuary.

Qualitative models of system dynamics are typically not specified in a way that ensures global conservation of real world quantities such as water, sediment or energy. It is, however, important to introduce a degree of rigour and consistency into the definition of the system components. The scheme adopted here considers the main morphological components to be volumetric features, some of which are mutually exclusive in a manner that is consistent with the spectrum of estuary types, and some of which can co-exist. Morphological changes can interact directly with tidal prism as the main internal state variable; some also feedback to influence wave energy. Sediment supply variables are envisaged to take the form of stocks, or background concentrations in essentially infinite ocean or coastal reservoirs. Other forcing factors are taken to imply the existence or otherwise of significant wind or wave energy to accomplish geomorphological work associated with their associated processes.

The interactions between the major sets of system variables are summarised in a set of influence graphs (or causal loop diagrams, in the sense of Puccia and Levins, 1985). At one level of abstraction, the interaction between outer and inner estuary sub-systems is depicted (Figure 7). More detailed influence graphs indicate the internal functioning of these sub-systems (Figure 8).

It is emphasised that these diagrams reflect the fact that the behaviour of the estuary system is known only in a very general sense and that each of the process and morphological variables is itself an abstraction of highly complex real world behaviour. Only a few of the interrelationships depicted here are well enough understood to permit the derivation of continuous mathematical functions or empirical scaling relationships. More often, what is being represented is merely accumulated experience derived variously from case studies and an appreciation of underlying processes studied at a small scale and in isolation. Translating such a general level of understanding into a form of simulation model that is amenable both to some form of mathematical implementation and to rigorous interpretation of the modelled system behaviour is probably the major biggest challenge in qualitative modelling (see, for example, Wolstenholme, 1999).

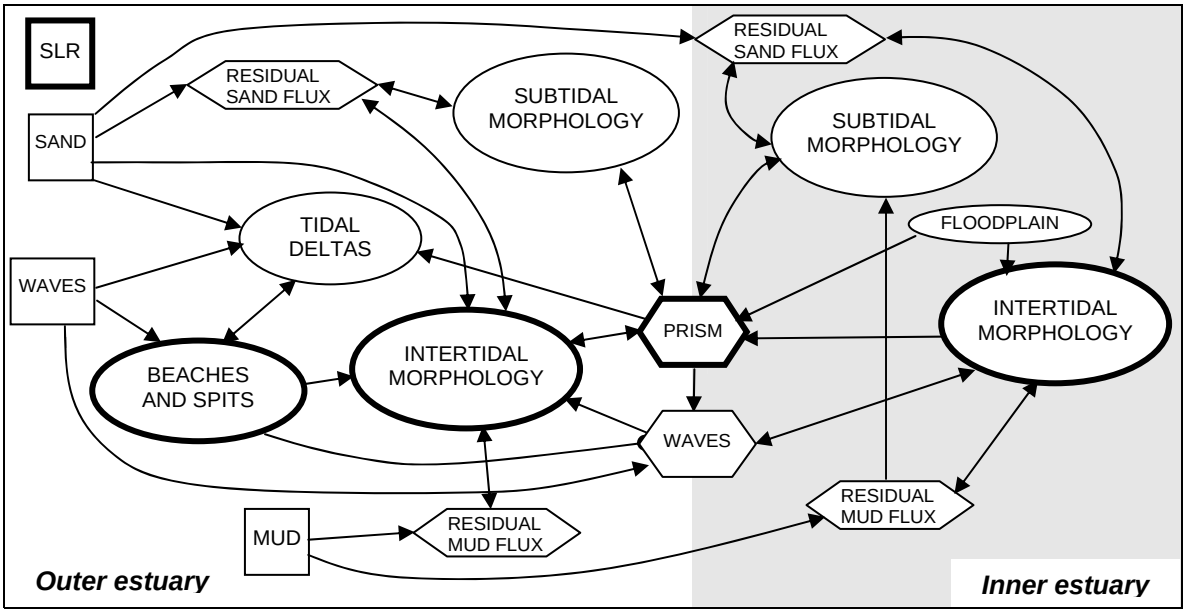


Figure 7: Global influence diagram for estuary comprising outer and inner sub-systems. Arrows indicate major indicative linkages between morphological and process variables, and bold outlines show most likely sensitivities to accelerated sea-level rise.

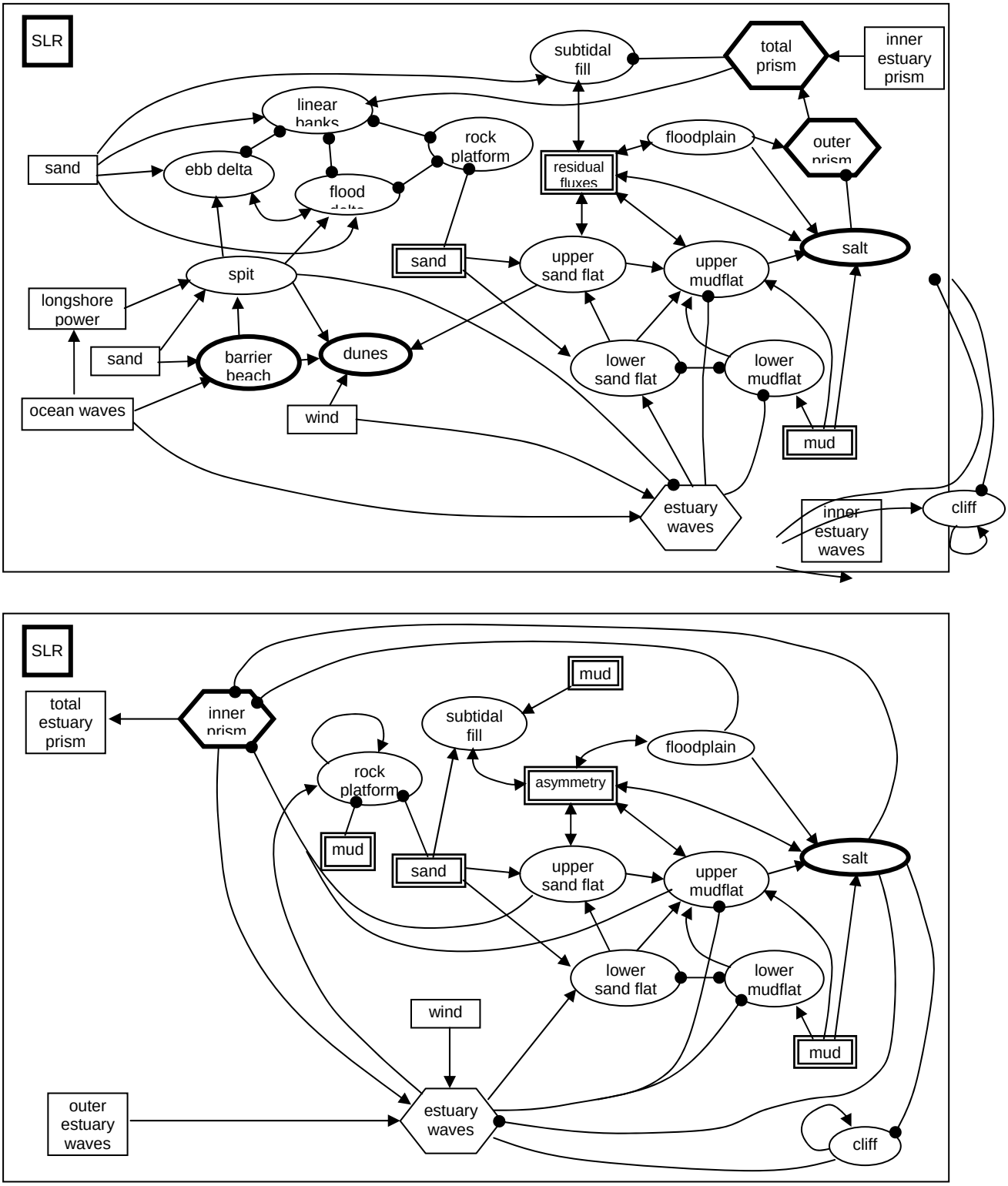


Figure 8: Influence diagram for outer (top) and inner estuary (bottom) sub-systems. Negative influences indicated by —●. Interaction between sand and mud supply and tidal asymmetry is not shown in detail. Fluvial inputs and anthropogenic forcing factors are omitted and some linkages are simplified for clarity.

Formulation of Boolean functions to characterise the linkages between them is a more challenging exercise and one that is likely to have a greater degree of 'operator variance', since formalisation of current scientific understanding is necessarily subjective. Translation from influence diagram to model equations remains a subjective and largely intuitive process, and there is currently no means of automatically converting an influence graph into a simulation model (see Wolstenholme, 1999; Smith, 2000). It follows that the Boolean functions derived below are not unique. Indeed variations will almost certainly result from interpretation of the influence diagram by different individuals (or 'experts'). Neither is it a trivial matter to ensure the logical consistency of the functions for anything but the simplest of models.

Tables 2 and 3 present a set of variables and associated Boolean functions based upon the preceding influence diagrams. System variables are indicated in lower case and are combined into associated Boolean functions, represented in upper case, defined using logical AND (& in *MATLAB*), OR (|) and NOT (~) operators. An EXCLUSIVE OR (xor) operator is also available and may also be used to improve the readability of the functions. Note that self-input is also allowed, such that source variables can represent imposed external forcing and connected variables representing resistant features can exist in imposed or inherited states. Single bit Boolean logic has been retained but model capability has been enhanced through the inclusion of additional state variables to handle gradational responses (e.g. for tidal prism, where something beyond low or high is useful).

The functions in Table 3 are not intended as a definitive set suitable for all applications. Rather, they represent a preliminary implementation of a Boolean network model in a manner that is consistent with the level of abstraction adopted in the formulation of the EstSim behavioural statements and system diagrams. The approach is extensible and further refinements could be implemented to take account of a broader consensus of expert opinion regarding the functional interaction of the various system components. In particular, it would be instructive to investigate the effect of differences in scientific opinion (or 'operator variance') on the formulation of the network functions and on the behaviour of the resulting systems.

Interconnections between the components of a Boolean system can be interpreted in terms of either existence or evolutionary tendency. The system representations that follow assume that Boolean states 0 and 1 refer to negligible or significant presence (in the limit, absence or presence), which allows a more consistent handling of process and morphological components than is possible when thinking in terms of tendency. Tendency can of course be inferred from the evolutionary behaviour of the system. In practice, such distinctions do not seem to be of critical importance provided that the implementation and terminology are consistent.

Table 2: Preliminary set of Boolean variables for EstSim model. Three additional state variables are used to provide finer resolution of total tide prism for the two sub-systems whilst keeping the functions as simple as possible (a simple truth table was used to define the functions). The extra variables could be eliminated through re-coding or else hidden at the output stage.

Variable name	Description
ow	Ocean waves {external forcing variable}
lwp	Longshore wave power {external forcing variable}
ms	Marine sand supply {external forcing variable}
mm	Marine mud supply {external forcing variable}
wd	Wind regime favourable for dunes {external forcing variable}
slr	Sea-level rise {external forcing variable}
dredge	Dredging of estuary channel {external forcing variable}
barrage	Barrage (with restricted tidal exchange) {external forcing variable}
oew	Outer estuary waves {state variable}
tp_o	Outer estuary prism {state variable}
ttp_l	Total estuary prism - low {accounting variable}
ttp_m	Total estuary prism - med {accounting variable}
ttp_h	Total estuary prism - high {accounting variable}
depth_ratio_o	Ratio of HW to LW mean depth - outer estuary {accounting variable}
area_ratio_o	Ratio of HW to LW area - outer estuary {accounting variable}
sg_o	Sand residual transport - outer estuary {state variable}
mg_o	Mud residual transport - outer estuary {state variable}
bb	Barrier beach
sp	Spit
du	Dune
ed	Ebb delta
fd	Flood delta
lb	Linear banks
ic	Incised outer estuary channel
osf_l	Sand flat - lower, outer estuary
osf_u	Sand flat - upper, outer estuary
omf_l	Mud flat - lower, outer estuary
omf_u	Mud flat - upper, outer estuary
osm	Saltmarsh - outer estuary
osubtidal_fill	Subtidal infill - outer estuary
orp	Rock platform - outer estuary
ofp	Tidal floodplain - outer estuary
ocf	Cliff - outer estuary
iew	Inner estuary waves {state variable}
tp_i	Inner estuary prism {state variable}
depth_ratio_i	Ratio of HW to LW mean depth - inner estuary {accounting variable}
area_ratio_i	Ratio of HW to LW area - inner estuary {accounting variable}
sg_i	Sand residual transport - inner estuary {state variable}
mg_i	Mud residual transport - inner estuary {state variable}
isf_l	Sand flat - lower, inner estuary
isf_u	Sand flat - upper, inner estuary
imf_l	Mud flat - lower, inner estuary
imf_u	Mud flat - upper, inner estuary
ism	Saltmarsh - inner estuary
isubtidal_fill	Subtidal infill - inner estuary
irp	Rock platform - inner estuary
ifp	Tidal floodplain
icf	Cliff
ri_q	River - high water discharge
ri_m	River - high mud load
ri_s	River - high sand load

Table 3: Boolean functions for variables for listed in Table 2. Self input is allowed, and some functions make a distinction between the initiation of a variable (i.e. the formation of a morphological feature) and its persistence (i.e. the maintenance of a feature). The underlying geomorphological logic is expanded in Appendix 3.

```
OW = ow & ~barrage
LWP = lwp & ow
MS = ((ms & ~slr) | (ms & ow)) & ~barrage
MM = mm
WD = wd
SLR = slr
DREDGE = dredge
BARRAGE = barrage

OEW = (ow & (~sp | ic)) | (wd & ~tp_o)
TP_O = (~osm & ofp) & ~barrage
TTP_L = ~(tp_o | tp_i)
TTP_M = xor(tp_o, tp_i)
TTP_H = tp_o & tp_i
DEPTH_RATIO_O = ~((osf_u | omf_u) & osubtidal_fill) | ~((osm | ofp))
AREA_RATIO_O = ~((osf_u | omf_u)  & ofp) | ~osm
SG_O = (depth_ratio_o & area_ratio_o) & ~ic
MG_O = ((osf_u | omf_u | ~tp_o)) | (mm & ~oew)
BB = (bb & ~slr) | (ow & ms)
SP = (bb & lwp & ms ) | (sp & bb & lwp & ~slr)
DU = (du & ~slr) | ((((bb | sp) & ms) | osf_u) & wd)
ED = ((ttp_l | ttp_m ) & ms & sp & ~orp & ~ic) | (ttp_l & ms & (bb | sp) & ~orp & ~ic)
FD = ((ttp_l | ttp_m ) & ms & sp & ~orp & ~ic) | (ttp_l & ms & (bb | sp) & ~orp & ~ic)
LB = (ttp_h | tp_o) & ms & ~(ed | fd | sp | orp | ic) & ~barrage
IC = ic
OSF_L = (ms & (sg_o & oew)) | (osf_l & ~(~ms & (oew | ~sg_o)))
OSF_U = (ms & oew & osf_l) | (osf_u & ~(~ms & (oew | ~sg_o)))
OMF_L = (  (mm & (mg_o | ~oew)) | (omf_l & ~(~mm & oew))) & ~osf_l
OMF_U = (mm & (mg_o & ~oew) & (omf_l | osf_l)) | (omf_u & ~(~mm & oew))
OSM = (omf_u & mm & (~oew & mg_o))
OSUBTIDAL_FILL = ((ms & sg_o) | (osubtidal_fill & ~(ofp & ~sg_o)) )  & ~(dredge | ri_q)
ORP = (orp & ~(osf_u | omf_u | osm)) | ( ~(osf_u | omf_u | osm) & oew & (~sg_o |  ~mg_o))
OFP = ofp
OCF = ocf & (oew | ~osm)

IEW = ((wd | oew) & tp_i) & ~ism
TP_I = (~(imf_u | isf_u | ism | ifp) & ~barrage)
DEPTH_RATIO_I = ~( (ifp | ism) & isubtidal_fill)
AREA_RATIO_I = ~((isf_u | imf_u)  & ifp) | ~ism
SG_I = ms & depth_ratio_i & area_ratio_i & ~ri_q
MG_I = (mm & (isf_u | imf_u | ~tp_i)) | (mm & ~iew)
ISF_L = (ms & sg_o & (sg_i | iew)) | (iew & ri_s) | (isf_l & ~(~ms & (iew | ~sg_i)))
ISF_U = (ms & sg_o & iew & isf_l) | (isf_u & ~(~ms & (iew | ~sg_i)))
IMF_L = ( ((mm & (mg_i | ~iew))) | (imf_l & ~(~mm & iew))) & ~isf_l
IMF_U = (mm & (mg_i | ~iew) & (imf_l | isf_l)) | (imf_u & ~(~mm & iew))
ISM = (((imf_u | isf_u) & mm & ~iew) | (ifp & mm & ~iew)) & ~(slr & ~ifp)
ISUBTIDAL_FILL = ((ms & sg_i) | (mm & mg_i) | (isubtidal_fill & ~(ifp & (~sg_i
     | ~mg_i))))  & ~(dredge | ri_q)
IRP = (irp & ~(isf_u | imf_u | ism)) | ( ~(isf_u | imf_u | ism) & iew & (~sg_i | ~mg_i |
     ri_q) )
IFP = ifp
ICF = icf & (iew | ~ism)
RI_Q = ri_q
RI_M = ri_m
RI_S = ri_s
```

6.3 Simulation of generic estuary types

The first test of the EstSim model is whether or not it is capable of discriminating between the seven generic estuary types defined in FD2117 Project Report 2 '*EstSim Behavioural Statements Report* 2' (EstSim Consortium, 2004). These are defined at a fairly high level of abstraction (see Appendix 4 for a summary of the main process and morphological components) and some further effort is needed to implement each type using the expanded variable set developed in the preceding section. Table 4 presents initial conditions for each type, with major forcing variables and morphological components essential to the estuary definition imposed. Remaining variables are free to evolve. The evolutionary behaviour of each initial state can now be considered.

Table 4: Initial conditions used for simulation of generic estuary types (external forcing and morphological response variables only).

Estuary type:	1	2	3	4	5	6	7
forcing							
ow	1	1	1	1	1	1	1
lwp	0	0	1	1	0	0	1
ms	0	0	1	1	1	1	1
mm	0	1	1	1	1	1	1
wd	1	1	1	1	1	1	1
outer estuary							
bb	0	0	0	0	0	0	0
sp	0	0	0	0	0	0	0
du	0	0	0	0	0	0	0
ed	0	0	0	0	0	0	0
fd	0	0	0	0	0	0	0
lb	0	0	0	0	0	0	0
ic	0	0	1	0	0	0	0
osf_l	0	0	0	0	0	0	0
osf_u	0	0	0	0	0	0	0
omf_l	0	0	0	0	0	0	0
omf_u	0	0	0	0	0	0	0
osm	0	0	0	0	0	0	0
osubtidal_fill	0	0	0	0	0	0	0
orp	1	1	0	0	0	0	0
ofp	0	0	0	0	0	0	0
ocf	1	1	1	0	0	0	0
inner estuary							
isf_l	0	0	0	0	0	0	0
isf_u	0	0	0	0	0	0	0
imf_l	0	0	0	0	0	0	0
imf_u	0	0	0	0	0	0	0
ism	0	0	0	0	0	0	0
isubtidal_fill	0	0	0	0	0	0	0
irp	1	1	0	0	0	0	0
ifp	0	0	0	0	0	0	0
icf	1	1	0	0	0	0	0
ri_q	1	1	1	1	1	0	0
ri_m	0	1	1	1	1	0	0
ri_s	1	1	1	1	1	0	0

Table 5: End states simulated for generic estuary types using initial conditions specified in Table 4. Morphological response and selected state variables shown. Values of 1 indicate significant presence or positive tendency (import in case of sediment transport). Cyclical end points shown in bold.

Estuary type:	1	2	3	**4**	5	6	**7**
oew	1	1	1	**1**	1	1	**1**
tp_o	1	1	1	**0**	1	1	**1**
sg_o	0	1	0	**1**	1	1	**1**
mg_o	0	0	0	**1**	1	1	**1**
bb	0	0	1	**1**	1	1	**1**
sp	0	0	1	**1**	0	0	**1**
du	0	0	1	**1**	1	1	**1**
ed	0	0	0	**1**	0	0	**1**
fd	0	0	0	**1**	0	0	**1**
lb	0	0	0	**0**	1	1	**0**
ic	0	0	1	**0**	0	0	**0**
osf_l	0	0	0	**1**	1	1	**1**
osf_u	0	0	0	**1**	1	1	**1**
omf_l	0	0	1	**0**	0	0	**0**
omf_u	0	0	0	**1**	0	0	**1**
osm	0	0	0	**1**	0	0	**1**
osubtidal_fill	0	0	0	**0**	0	1	**1**
orp	1	1	1	**0**	0	0	**0**
ofp	0	0	0	**0**	0	0	**0**
ocf	1	1	1	**0**	0	0	**0**
iew	1	0	0	**0**	0	0	**0**
tp_i	1	0	0	**0**	0	0	**0**
sg_i	0	0	0	**0**	0	0	**0**
mg_i	0	1	1	**1**	1	1	**1**
isf_l	1	0	0	**1**	1	1	**1**
isf_u	0	0	0	**1**	1	1	**1**
imf_l	0	1	1	**0**	0	0	**0**
imf_u	0	1	1	**1**	1	1	**1**
ism	0	1	1	**1**	1	1	**1**
isubtidal_fill	0	0	1	**0**	0	1	**1**
irp	1	0	0	**0**	0	0	**0**
ifp	0	0	0	**0**	0	0	**0**
icf	1	0	0	**0**	0	0	**0**
ri_q	1	1	0	**1**	1	0	**0**
ri_m	0	1	1	**1**	1	0	**0**
ri_s	1	1	1	**1**	1	0	**0**

6.3.1: Fjord

This is modelled as a sediment deficient system, with the exception of a supply of river sand (which allows some inner sand flats). Unsurprising, few depositional features form - there are no tidal deltas or linear banks; no intertidal sedimentary units (with the exception of low inner estuary flats); and tidal prism remains large. A steady state, characterised by active cliffs and rock platforms emerges after just 4 steps. Some fjords do have spits, yet an abundant supply of sand would lead to infilling of the accommodation space and conversion to another estuary type (such as a funnel-shaped estuary). However, spits might originate from reworking of relict sediment sources (e.g. offshore fluvial-glacial sands and gravels) and the associated Boolean function (Table 3) thus allows a spit to persist where a barrier beach exists, longshore transport is significant and sea-level rise is low (otherwise the feature will degenerate). The imposition of a spit does not affect the evolution towards the steady state end point referred to above.

6.3.2: Fjard

On the basis of the EstSim behavioural statements, the distinction between fjords and fjards is rather subtle. The former are over-deepened through glacial erosion and exhibit a shallower sill near the mouth; the latter typically contain rather more extensive intertidal deposits, including inner estuary saltmarsh. The bathymetric contrast is not presently represented (though some form of imposed bedrock topography could probably be implemented). The two types thus function in essentially the same way. However, the addition of a marine mud input (reflected in the initial conditions of Table 4) does cause an evolution towards a steady state (after 7 steps) in which the inner estuary contains mudflat and saltmarsh, whilst cliffs and rock platforms are retained in the outer estuary. The distinction between Fjards and Fjords probably needs a little more thought in terms of both the underlying estuary behavioural statements and the implementation in the model of inherited bathymetric characteristics and their interaction with contemporary physical processes.

6.3.3: Ria

Rias are slightly problematic in that they can potentially contain a range of intertidal depositional features, which implies a supply of sand and/or mud. However, the accommodation space remains substantially unfilled by Holocene sedimentation and their inherited fluvial planform contrasts with that of sediment-rich funnel-shaped (type 5) systems. This can be accommodated via the inclusion of an incised outer estuary channel, which precludes tidal delta formation and allows ocean wave propagation into the outer estuary, thereby reducing the protective influence of any spit that is present. Accordingly, deposition is largely restricted to the inner estuary, with inherited cliffs defining the planform of the outer estuary. Defined in this way, a steady state is reached after 8 steps (Table 5).

6.3.4: Spit-enclosed estuary

This system is forced with the imposition of a longshore wave power component that will lead to the formation of a protective spit (given a sediment supply). A river inflow is also imposed and distinguishes this type from a tidal inlet (type 7). The end point is cyclic between steps 10 and 16. Cyclic behaviour is confined to the outer estuary and involves interaction between saltmarsh and estuary waves. Essentially, however, the behaviour involves long-term infilling of both outer and inner estuary units, aided by the protection afforded by a spit. Ebb and flood deltas replace the linear banks of estuary types 5 and 6. The evolution is presented in full in Table 6.

6.3.5: Funnel-shaped drowned river valley

In this case, the estuary evolves to a steady state in 7 steps. The estuary shows a tendency to infill, with the outer and inner estuary intertidal areas being sand and mud-dominated respectively. Subtidal infilling is prevented by imposition of a significant river discharge, although this is not necessarily the case for all such estuaries. Dunes form in the outer estuary, given supply of sediment and imposition of a favourable wind regime. The outer estuary does not form tidal deltas, but is characterised by linear banks. The inner estuary saltmarsh leads to an ebb-dominated regime, although the outer estuary remains flood-dominant. Table 7 shows the full evolutionary sequence.

Table 6: Evolution of spit-enclosed estuary from minimal initial configuration towards cyclic equilibrium (indicated by shaded zone).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	step
0	1	1	0	0	0	0	0	1	1	1	0	0	0	1	1	oew
0	1	1	1	1	1	1	0	0	0	1	1	1	0	0	0	tp_o
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	sg_o
0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	mg_o
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	bb
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	sp
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	du
0	0	1	0	0	0	0	1	1	1	1	1	1	1	1	1	ed
0	0	1	0	0	0	0	1	1	1	1	1	1	1	1	1	fd
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	lb
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ic
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	osf_l
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	osf_u
0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	omf_l
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	omf_u
0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	osm
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	osubtidal_fill
0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	orp
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ofp
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ocf
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	iew
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	tp_i
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	sg_i
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	mg_i
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	isf_l
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	isf_u
0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	imf_l
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	imf_u
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	ism
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	isubtidal_fill
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	irp
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ifp
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	icf
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ri_q
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ri_m
1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	ri_s

6.3.6: Embayment

As presently defined in the EstSim project, embayments differ from the preceding funnel-shaped estuary type only in the lack any major river inflow. Given that the latter is not always significant in funnel-shaped systems, little difference in behaviour is to be expected. This is borne out by the model results, which show evolution towards a steady state characterised by outer linear banks and sand flats, and inner mudflats and saltmarsh. This evolution is accomplished in 8 steps.

Table 7: Evolution of funnel-shaped estuary towards steady state equilibrium.

1	2	3	4	5	6	7	step
0	1	1	1	1	1	1	oew
0	1	1	1	1	1	1	tp_o
0	0	1	1	1	1	1	sg_o
0	1	0	0	0	1	1	mg_o
0	1	1	1	1	1	1	bb
0	0	0	0	0	0	0	sp
0	0	1	1	1	1	1	du
0	0	1	0	0	0	0	ed
0	0	1	0	0	0	0	fd
0	0	1	0	0	0	1	lb
0	0	0	1	1	1	1	osf_l
0	0	0	0	1	1	1	osf_u
0	1	1	1	0	0	0	omf_l
0	0	0	0	0	0	0	omf_u
0	0	0	0	0	0	0	osm
0	0	0	0	0	0	0	osubtidal_fill
0	0	1	1	1	0	0	orp
0	0	0	0	0	0	0	ofp
0	0	0	0	0	0	0	ocf
0	0	1	1	0	0	0	iew
0	1	1	0	0	0	0	tp_i
0	0	0	0	0	0	0	sg_i
0	1	1	1	1	1	1	mg_i
0	0	0	1	1	1	1	isf_l
0	0	0	0	1	1	1	isf_u
0	1	1	1	0	0	0	imf_l
0	0	1	1	1	1	1	imf_u
0	0	0	0	0	1	1	ism
0	0	0	0	0	0	0	isubtidal_fill
0	0	0	0	0	0	0	irp
0	0	0	0	0	0	0	ifp
0	0	0	0	0	0	0	icf
1	1	1	1	1	1	1	ri_q
1	1	1	1	1	1	1	ri_m
1	1	1	1	1	1	1	ri_s

6.3.7: Tidal inlet

The key elements of a tidal inlet are the absence of any river flow and the tendency for one or more spits and tidal deltas at the estuary mouth. With the minimal set of initial conditions imposed in Table 4, such a system does indeed emerge. However, the end point is cyclic between steps 10 and 16 (see Table 8 for full evolutionary history). Cyclic behaviour is confined to the outer estuary and involves interaction between saltmarsh and estuary waves. Essentially, however, the behaviour involves long-term infilling of both outer and inner estuary units, aided by the protection afforded by a spit. Ebb and flood deltas replace the linear banks of estuary types 5 and 6.

Table 8: Evolution of tidal inlet towards cyclic equilibrium (indicated by shaded zone).

1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16	step
0	1	1	0	0	0	0	0	1	1	1	0	0	0	1	1	oew
0	1	1	1	1	1	1	0	0	0	1	1	1	0	0	0	tp_o
0	0	1	1	1	1	1	1	0	0	0	1	1	1	0	0	sg_o
0	1	0	0	1	1	1	1	1	1	1	1	1	1	1	1	mg_o
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	bb
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	sp
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	du
0	0	1	0	0	0	0	1	1	1	1	1	1	1	1	1	ed
0	0	1	0	0	0	0	1	1	1	1	1	1	1	1	1	fd
0	0	1	0	0	0	0	0	0	0	0	0	0	0	0	0	lb
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	osf_l
0	0	0	0	0	0	0	0	0	1	1	1	1	1	1	1	osf_u
0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	omf_l
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	omf_u
0	0	0	0	0	0	1	1	1	0	0	0	1	1	1	0	osm
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	osubtidal_fill
0	0	1	1	1	1	0	0	0	0	0	0	0	0	0	0	orp
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ofp
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ocf
0	0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	iew
0	1	1	0	0	0	0	0	0	0	0	0	0	0	0	0	tp_i
0	0	1	1	1	1	1	0	0	0	0	0	0	0	0	0	sg_i
0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	1	mg_i
0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	isf_l
0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	1	isf_u
0	1	1	1	0	0	0	0	0	0	0	0	0	0	0	0	imf_l
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	imf_u
0	0	0	0	0	1	1	1	1	1	1	1	1	1	1	1	ism
0	0	1	1	1	1	1	1	1	1	1	1	1	1	1	1	isubtidal_fill
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	irp
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ifp
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	icf
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ri_q
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ri_m
0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	0	ri_s

6.4 Statistical properties of generic estuary systems

Estuary systems defined in this way have a variable connectivity (K). Given that most of the variables are common to all of the generic types, there is little variation in mean K range (approximately 3.7 to 3.9). The median and maximum K are 4 and 8 respectively for all systems. This contrasts with most of the networks studied by theoretical biologists, in which K is generally fixed at a low value (K = 1 or 2). For fixed K = 2 networks it has been shown (e.g. Kaufman, 1996) that the number of system end points is approximately $\sqrt{N}$. Where N becomes very large, however, the number of end points appears to increase more rapidly than $\sqrt{N}$ (Bagley and Glass, 1996), and other studies have shown a larger proportion of stable states once irrelevant variables are excluded from the analysis (e.g. Bilke and Sjunnesson, 2002). Since the number of potential system states increases exponentially with N, the state variable space for the generic estuary systems must be investigated through statistical sampling of a small proportion of the possible initial conditions.

The estuary networks modelled here exhibit far more steady state and cyclical end points than would be predicted for fixed $K = 2$ systems. This is largely a consequence of the variable connectivity, K, (which adds structured complexity); the inclusion of a large number of source variables; and of the allowance of self-input. Statistical sampling (random sample of approximately 15,000 initial configurations) of a system incorporating all 48 variables used simulation of the generic estuary types identified over 3,200 unique end states. By no means have all of these correspond to meaningful configurations of the system being modelled. Most arise due to permutations of the source and state variables. It is sensible to constrain these during sampling such that external forcing variables are imposed (set to 1) and internal state variables initialised to 0 (i.e. left to evolve during individual runs). This greatly reduces the number of end states.

Table 9 presents a summary analysis of the state variable space for a generic tidal inlet (estuary type 7, $N = 38$), with varying degrees of constraint imposed upon the sampling of initial conditions. With all state variables initialised to zero, and random sampling of possible initial configurations of the remaining 23 variables (sample size approximately 0.13% of the potential 2^{23} initial states), a total of 852 unique end points are detected. Of these, about 18% are cyclic, with trajectory lengths of up to 16 and a maximum cycle length of 6 (which corresponds to the scenario defined explicitly in the preceding section). If external forcing variables are imposed (i.e. set to 1, and random initial configurations drawn from the remaining 2^{18} possible initial states), system behaviour becomes entirely cyclical and with just 56 unique end points detected.

Table 9: System behaviour statistics for a generic tidal inlet (estuary type 7) conceptualised in terms of 38 system components (with Boolean functions taken from Tables 3).

	Sampling constraint	
System parameter	State variables initialised to 0	State variables initialised to 0 Forcing variables initialised to 1
N	38	38
Effective N	23	18
K (median)	3.7	3.7
K (max)	6	6
Steady states	702 (82%)	0 (0%)
Cyclical states	150 (18%)	56 (100%)
Initial condition sample size	11,552	11,552
Unique system states utilised	29,235	14,878
Max trajectory length	16	15
Median trajectory length	4	10
Max cycle length	6	6
Median cycle length	6	6

Both steady state and cyclical endpoints can be thought of as attractors, with the set of initial conditions that lead to each attractor constituting its basin of attraction (Bagley and Glass, 1996). Summarising the number and type of attractor, and knowledge of how their basins of attraction partition the state space can provide important insights into how the system functions. Table 10

presents a summary classification of the attractors for the tidal inlet (estuary type 7) simulations presented above. It is notable that with the exclusion of most of the spurious stable states resulting from source variable permutations, system behaviour becomes organised into fewer and larger basins of attraction. In this particular system, all of the detected steady states are effectively spurious.

Even with constrained sampling of the initial conditions, the distribution of cyclical basin sizes remains skewed, with a large number of short cycles. Many of these are presumed to be artefacts arising from the synchronous updating used in the Boolean model. A similar point was made by Bagley and Glass (1996), who argued that one of the consequences of synchronous updating is to increase the number of cyclical end points compared to asynchronous systems. It is thus the larger basins of attraction that are of primary interest here and it is interesting that the basins detected through random sampling of the state space are similar in terms of cycle length to those that emerge through geomorphologically-informed experimentation with given initial conditions (section 6.3).

Table 10: Steady state and cyclic basins of attraction (number and size) for Estuarine Behavioural Type 7: generic tidal inlet.

	Constrained state variables		Constrained state variables Imposed forcing variables	
	steady	cyclical	steady	cyclical
Unique basins	702	150	0	56
median	17	21	0	84
mean	33.7	37.3	0	265.7
max	486	248	0	2041
min	3	6	0	11

6.5 EstSim model applied to historic change in the Ribble estuary

The Ribble estuary, northwest England, provides a useful test case for the application of the enhanced Boolean network model. The Ribble is a classic example of a funnel-shaped coastal plain estuary (EstSim estuary type 5), although a significant portion of the system has been reclaimed over the last 200 years (Figure 10). With regard to the 'inner – outer' spatialisation, the contemporary Ribble estuary comprises:

- outer estuary: sand flat (upper and lower), beach, dune, trained channel
- inner estuary: extensive saltmarsh (upper and lower), narrow sand and mud flats, trained channel

A series of detailed bathymetric charts span the last 150 years or so, and these have been used by van der Wal *et al.* (2002) to produce a synthesis of the morphological change over this period. The estuary is subject to a macro-tidal regime (mean spring tidal range of up to 8.0 m) and tidal flows are large compared to freshwater inflow from the River Ribble (and its tributary the River Douglas). Wave energy along the coast and in the outer estuary is moderate. Although fluvial and estuarine sediment sources are fairly minimal, the potential offshore supply is extremely large: the system is thought to be infilling in the main from this marine source, with additional alongshore contributions from the open coasts to the north and south.

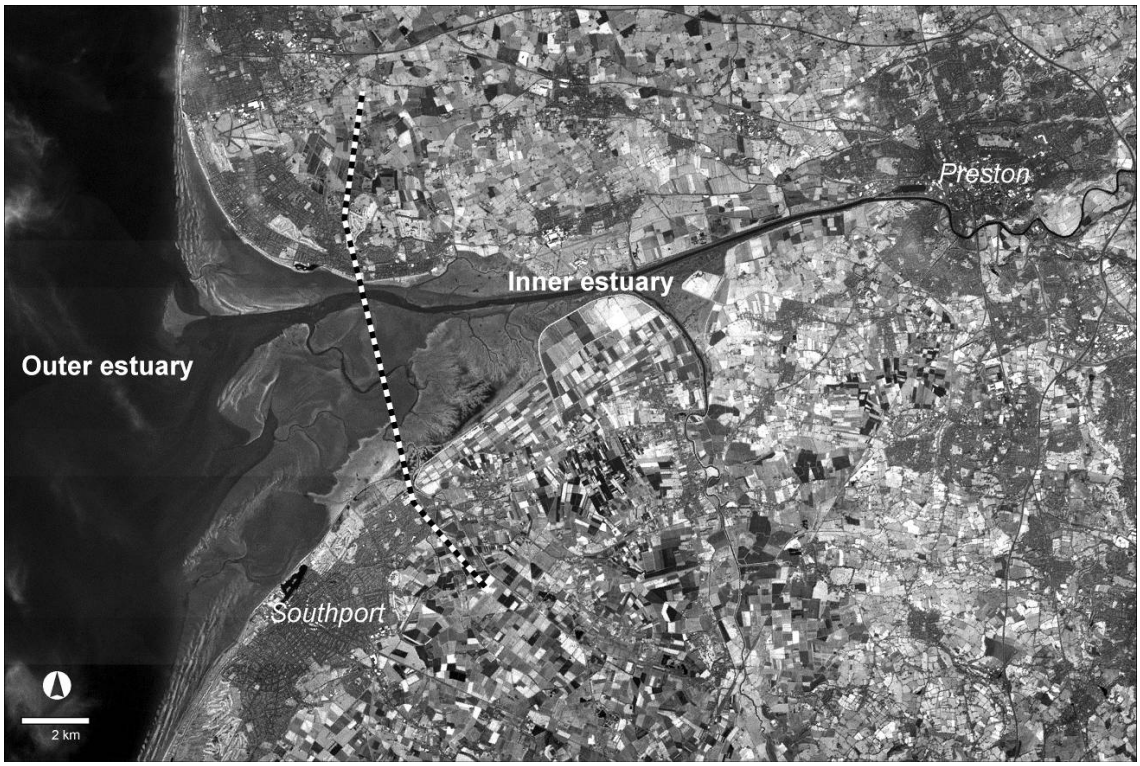


Figure 10: Delineation of outer and inner systems of the Ribble estuary (Landsat ETM, 2002).

Table 11 encapsulates the major changes in the morphology of the estuary since the 1840s into a series of time periods (corresponding to the main bathymetric surveys) and approximate system states, defined using the enhanced set of Boolean variables proposed in Section 6.2. The Ribble has experienced a number of human-induced perturbations to system dynamics over this historical period, including reclamation, construction of training walls and intermittent dredging, and is hence a useful case study for this modelling application. Subjective judgements are required in order to abstract the subtleties of the documented changes into a form that is commensurate with the resolution of the model.

In essence, Table 11 charts a transition from a rather wide system dominated by extensive low sand flats, to a narrower and rather deeper system which exhibits a clear transition from an intertidal sand-dominated outer estuary to mixed sediment, saltmarsh dominated inner estuary. Reclamation throughout the 19th century significantly reduced the tidal prism and intertidal area within the inner estuary: subsequent sedimentation, particularly during periods of dredging in the mid-20th century, preferentially infilled the reduced intertidal zone leading to an expanse of saltmarsh at the expense of intertidal flat. The channel in this region has been trained throughout the period considered here, and is now flanked by narrow strips of sand and mudflat. In the outer estuary, training walls constructed in the late 19th and early 20th century have replaced the multiple and dynamic character with a single, relatively stable channel, resulting in accretion across the intertidal flats. The cessation of dredging in the latter half of the 20th century has caused some local erosion of sand banks as the channel has started to accommodate some of the estuarine sediment volume.

Table 11: Summary of historic changes in the Ribble estuary, northwest England (largely adapted from van der Wal *et al.* (2002).

	Variables	1840s Description	1904 Description	1951 Description	1994 Description	Remarks
External forcing						
Ocean waves	ow_m	Moderate	Moderate	Moderate	Moderate	
Longshore wave power	lwp	Low	Low	Low	Low	
Marine sand supply	ms	High	High	High	High	
Marine mud supply	mm	High	High	High	High	
Wind regime for dunes	wd	Favourable	Favourable	Favourable	Favourable	
Sea-level rise	sl	Low	Low	Low	Low	
River inflow	ri_q, ri_m, ri_s	Low	Low	Low	Low	
Dredging	dredge	No	Yes	Yes	No	
System state variables						
Total tidal prism	otp_l, otp_m, otp_h	High	Medium	Medium	Low	reclamation, esp 1847 - 1904
Outer estuary waves	oew	Moderate	Moderate	Moderate	Moderate	
Inner estuary waves	iew	Moderate	Low	Low	Low	
Outer estuary morphology						
Channel	och_f, otp_n, otp_s	Wide, deep	Narrow, shallow	Narrow, deep	Narrow, shallow	post-1890 training, dredging, now ceased
Tidal delta	ed, fd, lb	Linear banks	None	None	None	
Sand flat	osf_l, osf_u	small, low	extensive low	extensive high	extensive high	sand flat replaced by mud flat, then marsh
Mud flat	omf	absent	absent	absent	absent	mud flat expands at expense of sand flat
Salt marsh	osm_l, osm_u	absent	absent	absent	small, low	reclamation, then planting and spread of Spartina
Beach	bb	Present	Present	Present	Present	
Dunes	du	Present	Present	Present	Present	
Spit	sp	None	None	None	None	
Rock platform	orp	None	None	None	None	
Inner estuary morphology						
Channel	ich_f, itp_n, itp_s	Narrow, deep	Narrow, shallow	Narrow, deep	Narrow, shallow	post-1890 training dredging, now ceased
Sand flat	isf	Extensive	Medium	Small	Absent	sand flat replaced by mud flat, then marsh
Mud flat	imf	Small	Small	Medium	Medium	mud flat expands at expense of sand flat
Salt marsh	ism_l & ism_u	Medium	Small	Medium	Extensive	reclamation, then planting and spread of Spartina
Tidal floodplain		Present	Absent	Absent	Absent	reclamation
Cliffs	cf	None	None	None	None	
Rock platform	irp	None	None	None	None	

Evolutionary end points predicted by the EstSim model represent long-term equilibrium configurations. An initial difficulty encountered during evaluation of model output against the documented change for the Ribble concerns the extent to which the real world system can be considered to have achieved equilibrium. Van der Wal *et al.* (2002) suggest that given the abundance of sediment and low rate of relative sea-level rise, the gross morphology of the Ribble was probably close to a state of dynamic equilibrium at the beginning of the 19th century. After that date, a succession of human interventions (reclamation, dredging, planting of saltmarsh, cessation of dredging) occurred at intervals that were probably much shorter than the relaxation time towards new equiluibrium states. It is likely, therefore, that we need to scrutinise not just the simulated end states but also their evolutionary trajectories. This needs to be done with care, being in mind the observation made in the previous section that some intermediate states are likely to be artefacts of the synchronous updating of the Boolean network.

As a first step, pre-1840 process forcing was approximated (high onshore wave energy, abundant sediment and no human intervention) and the model allowed to run to equilibrium (see Table 12). A steady state emerges after 6 steps. In contrast to the generic funnel-shaped estuary simulation in section 6.3, river inflow is assumed to be of negligible importance in this macro-tidal system. In reality, the small influx of river-derived silt was largely trapped within Preston Docks (van der Wal *et al*., 2002). Overall, the picture during this epoch is one of sedimentary infilling, with sandy intertidal deposits dominating the outer estuary (where wave action remains significant) and mud flat and saltmarsh being more widespread in the inner estuary. The general behaviour of this system in its pre-1840 state is thus captured very well.

Table 12: Simulated evolution of Ribble Estuary, for 1840 - 1904 epoch.

Summary of system evolution 1840 - 1904						
1	2	3	4	5	6	step
1	1	1	1	1	1	ow
1	1	1	1	1	1	ms
1	1	1	1	1	1	mm
1	1	1	1	1	1	wd
0	0	0	0	0	0	slr
0	0	0	0	0	0	dredge
0	1	1	1	1	1	oew
0	1	1	1	1	1	tp_o
0	0	1	1	1	1	sg_o
0	1	0	0	0	1	mg_o
0	1	1	1	1	1	bb
0	0	0	0	0	0	sp
0	0	1	1	1	1	du
0	0	1	0	0	0	ed
0	0	1	0	0	0	fd
0	0	1	0	1	1	lb
0	0	0	1	1	1	osf_l
0	0	0	0	1	1	osf_u
0	1	1	1	0	0	omf_l
0	0	0	0	0	0	omf_u
0	0	0	0	0	0	osm
0	0	0	1	1	1	osubtidal_fill
0	0	0	0	0	0	ofp
0	0	0	0	0	0	iew
0	0	0	0	0	0	tp_i
0	0	1	1	0	0	sg_i
0	1	1	1	1	1	mg_i
0	0	0	1	1	1	isf_l
0	0	0	0	0	0	isf_u
0	1	1	1	0	0	imf_l
0	0	1	1	1	1	imf_u
0	1	1	1	1	1	ism
0	0	1	1	1	1	isubtidal_fill
1	1	1	1	1	1	ifp

The major human interventions during the 19th century were reclamation of part of the inner estuary intertidal (with a reduction of the tidal floodplain and the area of saltmarsh), and dredging and training of the main estuary channel. These changes were used to create a set of modified boundary conditions and the model was re-run to simulate change over a period corresponding to the 1904 - 1951 epoch in Table 11. The result (Table 13) is a rapid return to steady state after just three steps, with saltmarsh returning to the inner estuary, and a removal of the subtidal deposits as a direct result of the dredging. This is not entirely consistent within the documented inner estuary changes, which involved a rather slower transition back to a muddy intertidal and re-establishment of saltmarsh accelerated by planting after the 1950s.

Table 3: Simulated evolution of Ribble Estuary, for 1904 - 1951 epoch.

```
Summary of system evolution 1904 - 1951 (modified 1904 input)
1  2  3  step

1  1  1  ow
1  1  1  ms
1  1  1  mm
1  1  1  wd
0  0  0  slr
1  1  1  dredge
1  1  1  oew
1  1  1  tp_o
1  1  1  sg_o
1  1  1  mg_o
1  1  1  bb
0  0  0  sp
1  1  1  du
0  0  0  ed
0  0  0  fd
1  1  1  lb
1  1  1  osf_l
1  1  1  osf_u
0  0  0  omf_l
0  0  0  omf_u
0  0  0  osm
1  0  0  osubtidal_fill
0  0  0  ofp
0  0  0  iew
0  0  0  tp_i
0  0  1  sg_i
1  1  1  mg_i
1  1  1  isf_l
0  0  0  isf_u
0  0  0  imf_l
1  1  1  imf_u
0  1  1  ism
1  0  0  isubtidal_fill
0  0  0  ifp
```

The final epoch sees the cessation of dredging, the effect of which is simply to allow subtidal infilling under a positive sediment gradient. This can be assumed to be sand deposition in the outer estuary, with muddier sediments being deposited in the inner estuary, which becomes ebb-dominated (in terms of velocity asymmetry) in the model.

Table 14: Simulated evolution of Ribble Estuary, for 1951 - 1994 epoch.

```
Summary of system evolution 1951 - 1994 (modified 1951 input)
1  2  3  4  step

1  1  1  1  ow
1  1  1  1  ms
1  1  1  1  mm
1  1  1  1  wd
0  0  0  0  slr
0  0  0  0  dredge
1  1  1  1  oew
1  1  1  1  tp_o
1  1  1  1  sg_o
1  1  1  1  mg_o
1  1  1  1  bb
0  0  0  0  sp
1  1  1  1  du
0  0  0  0  ed
0  0  0  0  fd
1  1  1  1  lb
1  1  1  1  osf_l
1  1  1  1  osf_u
0  0  0  0  omf_l
0  0  0  0  omf_u
0  0  0  0  osm
0  1  1  1  osubtidal_fill
0  0  0  0  ofp
0  0  0  0  iew
0  0  0  0  tp_i
1  1  1  0  sg_i
1  1  1  1  mg_i
1  1  1  1  isf_l
0  0  0  0  isf_u
0  0  0  0  imf_l
1  1  1  1  imf_u
1  1  1  1  ism
0  1  1  1  isubtidal_fill
0  0  0  0  ifp
```

The above simulations assume a dynamic equilibrium with a low rate of relative sea-level rise. One would not necessarily expect a sediment-rich system such as the Ribble to be particularly sensitive to accelerated sea-level rise. In fact, imposition of sea-level rise on the 1994 steady state has the effect of removing the inner estuary saltmarsh through 'coastal squeeze'. This also has the effect of restoring a flood dominant regime. There are no other morphological changes and a new steady state emerges after just 3 additional steps.

Overall, these results are encouraging in that they reproduce the gross evolutionary behaviour of the Ribble. Human intervention has been quite significant in this system, but a combination of abundant sediment supply and significant wave and tidal action mean that morphological equilibrium is readily re-established (though the absolute timescales for this are not specified). Some details are missed, and refinement of the inner estuary sub-system model and the role of fluvial inputs would be required in order to more fully investigate the impact of historic changes in the management of the docks and their approach channels.

6.5 Towards the development of a GUI-based tool

The prototype simulator has been implemented in MATLAB and presently supports:

- representation of estuary system diagrams for the generic estuary types defined in EstSim Project Report 2 using standardised sets of network variables and pre-defined Boolean functions

- state variable space simulation, based on either analysis of all possible states (small N systems) or statistical sampling of these states (larger N systems), that can be used to identify and classify equilibrium states, and derives various measures of system complexity

- initial condition-based simulation of system evolutionary trajectories

This implentation has the advantage that the Boolean functions are evaluated directly as mathematical expressions by *MATLAB*, thereby avoiding the need for the user to learn any form of programming language. All that is required is an appropriate conceptualisation of the system based upon a set of components (as specified in an estuary definition file, which can be supported by a linked descriptive behavioural statement) and their functional interdependencies (specified as a library of Boolean functions). The latter are presently defined as a universal set for all estuary types, but could be customised for individual estuary types or specific management problems.

Currently, a text format steering file is used to define one or more initial condition-based scenarios (an example is given in Appendix 5). However, we are developing a more interactive interface using *MATLAB's* GUI tools. Figure 11 illustrates a possible layout for the GUI and its relation to underlying model code, the Boolean function library and a set of estuary definition files. The latter may be associated with a database of behavioural statements, which could also include system diagrams. Once developed, the GUI-based tool will be compiled into an easy to use application that can be freely distributed to a wider range of users who do not have access to MATLAB software.

A simple standalone tool would supplement rather than replace a web-based simulator of the kind proposed in section 5. It might also provide a basis for more powerful software that supported advanced graphical features, such as animated system diagrams or virtual landscape visualisation.

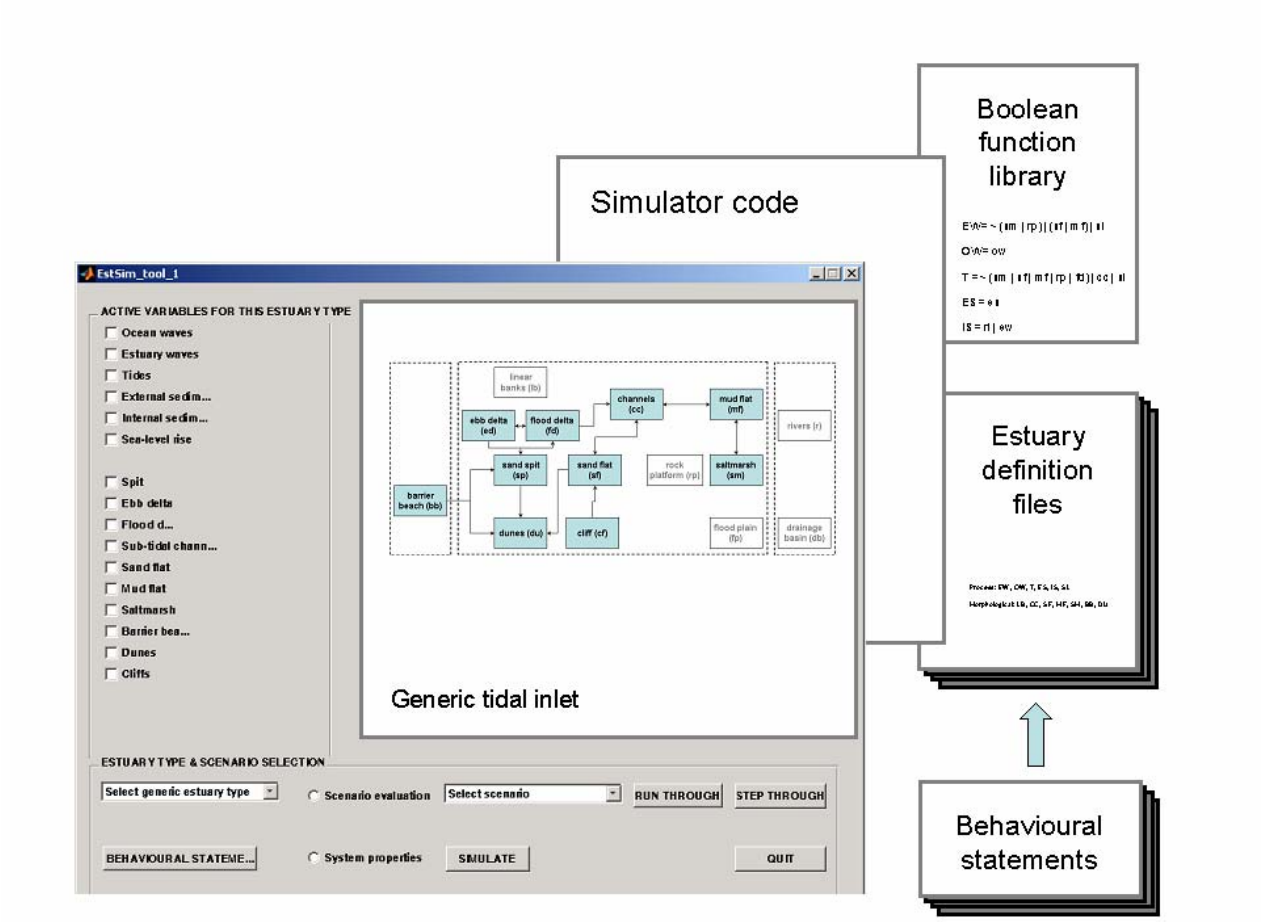


Figure 11: Illustration of possible GUI for prototype standalone EstSim tool developed in MATLAB, showing aspects of underlying architecture and functionality.

7. Conclusions and recommendations

A qualitative approach provides an alternative basis for modelling the behaviour of estuary (and coastal) systems, as abstracted at the scales of most relevance to the tackling of estuary management questions. Emergent properties of system behaviour may be revealed, including the existence of unexpected sensitivities to change and/or constraints upon the evolution towards morphological equilibrium. Qualitative judgements concerning estuary behaviour also provide a basis for evaluating the output from quantitative models (i.e. in terms of the direction of predicted change rather than its magnitude). Such insights can be crucial in impact and vulnerability assessment.

Of the various qualitative simulation approaches available the Boolean network formulation is particularly straightforward to implement. It is easy to program and presents no serious computational challenges, yet has the potential to predict important aspects of estuary system behaviour, including the existence of multiple equilibrium states and the qualitative response of a given stable state to environmental change (sea-level rise, sediment supply changes etc.) or interventions such as dredging. Moreover, the ability to work directly from symbolic Boolean functions avoids the need for complex graphical or algorithmic programming.

Two distinct kinds of software have been reviewed in terms of their suitability for qualitative system modelling and for the implementation of a Boolean network model in particular. These include, firstly, packages that allow the user to construct graphical representations of the system of interest before interactively specifying functional linkages and dependencies, and, secondly, numerical/visualisation software that requires the user to implement underlying model algorithms in some form of computer code.

Alternative architectures for a prototype estuary system simulator software tool have been considered. The most promising option appears to be an easy to use web-based decision support system comprising a browser-based interface to the EstSim classification of UK estuaries; their generic system diagrams; behavioural statements for their associated sub-systems; and a database of pre-computed system evolution trajectories with accompanying textual descriptions, linked to a menu of estuary types and most likely management questions. This could be used in conjunction with a simple GUI-based simulation tool (to be downloadable as freeware) that would include pre-defined system diagrams and a set of pre-defined scenarios, but would also allow a more experienced user to specify custom system diagrams or scenarios through external (open-format) run control or steering files.

Important issues concerning the selection of a scale of system abstraction that is commensurate with the assumptions made in the qualitative model have been identified. Great care is needed in the formalisation of existing geomorphological knowledge into logical and consistent systems diagrams. Also, the assumption that functional linkages take the form of a threshold relationship is not always justified. One way around this problem is an expansion of the variable set to distinguish between forcing, state and morphological response variables and to accommodate intermediate states. These issues are not unique to a Boolean network approach, although the adoption of single-bit logic with synchronous updating of system states imposes some restrictions on what can be modelled at a given level of abstraction.

A prototype model, implemented in MATLAB, shows the potential of the Boolean network approach to simulate the behaviour of idealised estuary systems. Superficially simple estuary systems are shown to exhibit potentially complex behaviour, including multiple steady state and oscillatory equilibrium states, and quite long evolutionary trajectories when initialised at a state of disequilibrium. Encouragingly, the seven generic estuary types defined in this project are discriminated by sensible combinations of forcing factors and morphological components. However, some additional work is needed to adapt the estuary behavioural statements so that these are more consistent with the mathematical formulation of the model, and to distinguish more effectively between fjords, fjards and rias.

Evaluation of the prototype model against a dataset of historic changes in the Ribble estuary yields broadly satisfactory results. The gross evolutionary behaviour of this funnel-shaped estuary is modelled reasonably well by a scheme incorporating outer and inner estuary sub-systems, although abstraction of historic data into a form that is commensurate with the model requires some care. Finer details of the kind that might be relevant to management questions, such as the impact of dredging or reclamation, are not so well resolved by a generic simulator implemented at this scale. However, the methodology presented here is extensible both in terms of the variable set and in the number of linked sub-systems.

References

Barabási, A-L., Albert, R. and Jeong, H. (1999) Emergence of scaling in random networks. *Science* 286, 510-12..

Bennett, R.J. and Chorley, R.J. (1978) *Environmental systems: philosophy, analysis and control*. London, Methuen, 624pp.

Bilke, S. and Sjunnesson, F. (2002) Stability of the Kauffman model. *Physical Review* E 65,

Capobianco, M., de Vriend, H., Nicholls, R. and Stive, M.J.F. (1999) Coastal area impact and vulnerability assessment: the point of view of a morphodynamic modeller. *Journal of Coastal Research* 15, 701-16.

Cowell, P.J. and Thom, B.G. (1994), Morphodynamics of coastal evolution. In: Carter, R.W.G. and Woodroffe, C.D. (eds.) *Coastal evolution: late quaternary shoreline morphodynamics*. Cambridge, Cambridge University Press, 33-86.

Dalrymple, R.W., Zaitlin, B.A., and Boyd, R. (1992) Estuarine facies model: conceptual basis and stratigraphic implications. *Journal of Sedimentary Petrology* 62, 1130-46.

de Vriend, H.J., Capobianco, M., Chesher, T., de Swart, H.E., Latteux, B. and Stive, M.J.F. (1993) Approaches to long-term modelling of coastal morphology: a review. *Coastal Engineering* 21, 225-69.

Dronkers, J. (1986) Tidal asymmetry and estuarine morphology. *Netherlands Journal of Sea Research* 20, 117-31.

EMPHASYS Consortium (2000) *Modelling estuary morphology and process*. Final Report for MAFF Project FD1401. HR Wallingford, Report TR111.

EstSim Consortium (2004) FD2117 Project Report 2 *EstSim Behavioural Statements Report*, 127pp.

Ford, A. (1999) *Modeling the environment. An introduction to system dynamics modeling of environmental systems*. Washington, Island Press, 415pp.

Fox, J.J. and Hill, C.C. (2001) From topology to dynamics in biological networks. *Chaos* 11, 809-15.

French, J.R., Reeve, D.E. and Owen, M. (2002) *DEFRA/EA Estuaries Research Programme – Phase 2 Research Plan*. London, DEFRA, 47pp.

Glass, L. (1975) Classification of biological networks by their qualitative dynamics. *Journal of Theoretical Biology* 54, 85-107.

Hibma, A., Stive, M.J.F., and Wang, Z.B. (2004) Estuarine morphodynamics. *Coastal Engineering* 51, 765-78.

Karunarathna, H. and Reeve, D. (2005) FD2117 Project Report 3 *Mathematical Formulation of Estuary Simulator*, 13pp.

Kaufman, S.A. (1996) *At home in the universe: the search for laws of self-organization and complexity*. Oxford, Oxford University Press.

Mulligan, M. and Wainwright, J. (2004) Modelling and model building. In: Wainwright, J. and Mulligan, M. (eds.) *Environmental modelling: finding simplicity in complexity*. Chichester, Wiley, 7-73.

Nicolis, C.(1982) A Boolean approach to climate dynamics. *Quarterly Journal of the Royal Meteorological Society* 108, 707-15.

O'Brien, M.P. (1969) Equilibrium flow areas of inlets on sandy coasts. *Journal of the Waterways and Harbours Division, American Society of Civil Engineers* 95, 43-52.

Odum, H.T. and Odum, E.P. (2000) *Modelling for all scales: an introduction to system simulation*. London, Academic Press.

Phillips, J.D. (1992) Nonlinear dynamical systems in geomorphology: revolution or evolution. *Geomorphology* 5, 219-29.

Saunders, A. and Ghil, M. (2001) A Boolean delay equation model of ENSO variability. *Physica D* 160, 54-78.

Sklar, F.H., Gopu, K.K., Maxwell, T. and Costanza, R. (1994) Spatially explicit and implicit dynamic simulations of wetland processes. In: W.J. Mitsch (ed.) *Global wetlands: old world and new*. New York, Elsevier, 537-54.

Townend I.H. (2003) Coast and estuary behaviour systems. In: Davis, R.A. (ed.) *Proceedings of the International Conference on Coastal Sediments 2003*. CD-ROM published by World Scientific Publishing Corp. and East Meets West Productions, Corpus Christi, Texas, USA, 13pp.

Townend, I. (2005) An examination of empirical stability relationships for UK estuaries. *Journal of Coastal Research*, 21, 1042-53.

van der Wal., D., Pye, K., and Neal, A. (2002) Long-term morphological change in the Ribble Estuary, northwest England. *Marine Geology* 189, 249-66.

von Bertalanffy, L. (1968) *General system theory: foundations, development and applications.* New York, George Braziller, 289pp.

Wright, L.D. and Thom, B.G. (1977) Coastal depositional landforms: a morphodynamic approach. *Progress in Physical Geography* 1, 412-59.

Appendix 1: GUI-based dynamic system simulation software product overview

Stella : isee systems, inc
Stella is one of the best-established Graphical User Interface (GUI)-based system modelling tools. Originally developed in 1987, it has evolved into a widely used tool for the modelling of a diverse range of system modelling problems across the natural and social sciences. A recent addition to the range is *NetSim Creator* that allows the creation of web-based interactive simulations that utilise a server-based installation of *Stella*.

Main features highlighted in marketing:

- intuitive icon-based graphical interface simplifies model building
- stock and flow diagrams support the common language of Systems Thinking and provide insight into how systems work
- enhanced stock types enable discrete and continuous processes with support for queues, ovens, and enhanced conveyors
- Causal Loop Diagrams present overall causal relationships
- model equations are automatically generated and made accessible beneath the model layer
- built-in functions facilitate mathematical, statistical, and logical operations
- arrays simply represent repeated model structure and sub-models support hierarchical model structures
- sensitivity analysis reveals key leverage points and optimal conditions
- results presented as graphs, tables, animations, QuickTime movies, and files
- dynamic data import/export links to Microsoft® Excel
- save as Runtime option creates full-screen, runtime models
- multimedia support for graphics, movies, sounds, and text messages based on model conditions

standalone models: Yes. Runtime models can be used by users with no access to *Stella* software.

open source: No, but models are saved in an open-format.

required expertise: Low-Medium (eliminates requirement for programming skills, but requires some experience and understanding of system simulation principles)

platform: Windows and MacOS-X systems

cost/licensing: single licence US$1899 for commercial users and US$649 for academic users (UK price: December 2005). Volume discounts available.

web: isee systems : www.iseesystems.com

ModelMaker : ModelKinetix

ModelMaker is an inexpensive and easy to use Graphical User Interface (GUI)-based system modelling tool. It is designed for scientists who are not necessarily expert mathematicians, programmers or modellers. Using a compartmental method of modelling users can quickly convert a conceptual model into a working mathematical model of a system of interest. The package is intended for application to problems in environmental, biochemical, geological, ecological or pharmacological science.

Main features highlighted in marketing:

- GUI-based system diagram construction
- extensive set of mathematical expressions for formulation of system relationships (including Boolean operations)
- modelling of continuous and discontinuous functions, stiff systems and stochastic systems
- 5 different integration methods - Runge-Kutta, Mid-Point, Euler, Bulirsch-Stoer and Gear
- optimisation and Monte Carlo simulation capabilities
- graphic and tabular output of results, with support for cut and paste into other applications

standalone models: No, although model files can be distributed amongst other *ModelMaker* users.

open source: No, and models appear to be stored in a closed proprietary format. No obvious means of integrating models with other applications.

required expertise: Low (eliminates requirement for programming skills, and is one of the simpler graphical model building and system simulation packages on the market. However, some understanding of system simulation principles is still needed in order to realise all of its capabilities)

platform: Windows systems

cost/licensing: single licence £210 for commercial users and £160 for academic users (UK price: December 2005). Printed documentation is available at extra cost. Volume discounts available.

web: ModelKinetix : www.modelkinetix.com/modelmaker

Simile : Simulistics Ltd

Simile is a relatively new addition to the range of Graphical User Interface (GUI)-based system modelling tools. Unlike many of its competitors, *Simile* is explicitly intended for the modelling of earth, environmental and life science systems. It uses a logic-based declarative modelling approach to represent the interactions with environmental systems in a structured and visually intuitive way.

Main features highlighted in marketing:

- GUI-based system diagram construction
- formal systems notation for model conceptualisation
- extensive set of mathematical expressions for formulation of system relationships
- support for matrix operations
- support for hierarchical model-building (including separate execution of sub-models)
- object-oriented constructs allow extremely concise representation of model structure
- completed models can be simulated using compiled C++ code – improving execution speed and allowing the incorporation of *Simile*-developed models into other applications
- extensive visualisation and presentation capabilities
- *Simile* models are stored in an intuitive open (ASCII text) pseudo-code format, which makes the modelling process more transparent to users

standalone models: No pre-packaged runtime viewer provided, but user-developed applications using the open source Tool Command Language and GUI Tool Kit (Tcl/Tk) available at no cost at www.tcl.tk .

open source: No, but Simile is very 'open-ended' in its implementation. Models are stored in portable open format and can be readily embedded in other applications developed using C++ or Tcl/Tk.

required expertise: Low-Medium (eliminates requirement for programming skills, but nonetheless a powerful package that requires some experience and technical understanding of system simulation principles in order to leverage all of its capabilities)

platform: Windows, MacOS-X and x86-linux systems

cost/licensing: single licence US$495 ($995 for developer licence) for commercial users (UK price: December 2005). Free evaluation version capable of handling models of restricted size.

web: Simulistics Ltd : www.simulistics.com

Vensim : Ventana Systems, Inc
Vensim is a visual modelling tool that allows conceptualisation, simulation and analysis of dynamic systems. It provides a simple and flexible way of building simulation models from causal loop or 'stock and flow' system diagrams By connecting words with arrows, relationships among system variables are entered and recorded as causal connections. This information is used by an interactive equation editor to complete the simulation model.

Main features highlighted in marketing:

- GUI-based system diagram construction
- equation editor for formulation of system relationships
- extensive library of built-in functions
- support for optimisation, sensitivity analyses and Monte Carlo simulation
- support for external functions (e.g. coded in C++)
- the *Vensim* DLL allows control of *Vensim* from Visual Basic, Delphi or any other programming language
- extensive visualisation and presentation capabilities

standalone models: Yes, packaged applications and complied models (C language) can be generated by more expensive versions.

open source: No. Compiled model files stored in proprietary binary format.

required expertise: Low-Medium (eliminates requirement for programming skills, but nonetheless a powerful package that requires some experience and technical understanding of system simulation principles in order to leverage all of its capabilities)

platform: Windows and MacOS-7 (or later) systems

cost/licensing: single licence US$129 to $1995 (depending on version / functionality) for commercial users (UK price: December 2005). Academic and volume discounts are available. A limited functionality educational version is available as freeware.

web: Ventana Systems, Inc : www.vensim.com

VisSim : Visual Solutions, Inc
VisSim is Graphical User Interface-driven software based around the *VisSim* visual block diagram language for modelling and simulation of complex nonlinear dynamic systems. The software combines an intuitive drag-and-drop block diagram interface with a powerful simulation engine. The visual interface offers a simple method for constructing, modifying, and maintaining complex system models. Its tightly integrated development platform makes it easy to pass freely between model construction, simulation, optimization, and validation, without having to undertake any traditional computer programming.

Main features highlighted in marketing:

- Drag-and-drop block diagram construction
- 110+ linear and nonlinear blocks (including Boolean functions)
- Toolbox functions for control, electromechanical design, hydraulics, signal processing etc.
- Euler, trapezoidal, Runge Kutta 2nd and 4th orders, adaptive Bulirsh-Stoer and Runge Kutta 5th order, stiff backward Euler integration algorithms
- Implicit system solvers
- Vector and matrix operations
- Hierarchical models with embedded sub-diagrams
- 'What-if'scenarios
- Parameter optimization
- Synchronous or asynchronous data exchange
- DLL wizard for custom C, C++, Fortran, and Pascal blocks
- *MATLAB, MathCAD*, and *Maple* integration

standalone models: Yes. Models can be developed and distributed for use with freeware *VisSim* model viewer.

open source: No. *VisSim* model 'scripts' are distributable in open ASCII text format.

required expertise: Medium (eliminates requirement for programming skills, but nonetheless a very powerful package that requires experience and technical understanding of system simulation principles in order to leverage all of its capabilities)

platform: Windows systems

cost/licensing: single licence £1995 for commercial users (UK price: December 2005). Discounts available for academic users.

web: Visual Solutions, Inc : www.vissim.com

Extend : Imagine That, Inc
Extend is designed to be a flexible, extendable simulation tool that is primarily intended for the modelling of business organisations and their associated processes. Like other system modelling software, it includes an interactive and graphical architecture that is integrated with a powerful and robust model development environment.

Main features highlighted in marketing:

- GUI-based system 'drag and drop' model construction
- extensive set of mathematical expressions for formulation of system relationships
- support for matrix operations
- support for hierarchical model-building
- full suite of interprocess tools for communicating with other applications
- extensive visualisation and presentation capabilities
- *Extend* models are stored in open (ASCII text) format

standalone models: Yes, using freeware viewer, although this has no print or save functionality. A run time-only version of the software is available at nominal cost.

open source: No, but *Extend* is 'open-ended' in that model 'block components' are stored in a open format to allow modification and enhancement. Users can thus alter existing blocks and develop new proprietary components. Linking to code and routines written in external languages (e.g. C, Fortran) is also supported.

required expertise: Medium (eliminates requirement for programming skills, but nonetheless a powerful package that requires experience and technical understanding of system simulation principles in order to leverage all of its capabilities)

platform: Windows, MacOS-X systems

cost/licensing: single licence US$895 to US$1595 depending on version (December 2005). Complex licensing options, include a run time only licence that costs US$95 and a freeware 'model player'. Discounts available for academic users.

web: Imagine That, Inc : www. imaginethatinc.com

PowerSim : PowerSim Software AS
PowerSim provides a powerful environment for the conceptualisation and modelling of systems, using graphical model building tools that allow for the construction of models according to formal principles of systems theory, and simulation capabilities that allow deductions to be made concerning the behaviour of systems so defined. The software is primarily intended and marketed for use in the modelling of business organisations and industrial process operations. However, it is clearly capable of application to environmental systems.

Main features highlighted in marketing:

- GUI-based system diagram construction
- construction of models based on a formal systems dynamics paradigm
- A graphical modelling language, with a mathematical definition language similar to common spreadsheets
- support for logical operations
- support for units of measurement
- support for hierarchical model-building
- powerful presentation capabilities, with easy to use linking capabilities
- optimisation and sensitivity analysis
- data connectivity other applications (e.g. Microsoft *Excel*)

standalone models: Yes, using freeware *PowerSim* player.

open source: No. However, models can be distributed in an open format such that algorithms employed are accessible to other user of *PowerSim* (or *PowerSim* Player)

required expertise: Medium (eliminates requirement for programming skills, but nonetheless a powerful package that requires experience and technical understanding of system simulation principles in order to leverage all of its capabilities)

platform: Windows systems

cost/licensing: single licence £850 to 1750 (depending on version) for commercial users (UK price: December 2005). Academic development pack is £750. Significant discounts are available for multiple purchases.

web: Powersim Software AS : www.powersim.com

Appendix 2: Numerical computation and visualisation software product overview

MATLAB : The Mathworks, Inc

MATLAB integrates a high-level programming language with an extensive library of functions and other tools for data analysis, visualization, and cross-platform application development. The flexible MATLAB environment incorporates more than 1000 mathematical, statistical, and engineering functions, and interactive graphical capabilities for creating plots, images, surfaces, and volumetric representations. Add-on toolbox algorithms enhance functionality in areas such as signal and image processing, data analysis, and system modelling (notably through the *SIMULINK* toolbox).

Main features highlighted in marketing:

- powerful high-level language for numerical computation and data analysis
- extensible and customisable through user-defined functions
- a very large library of mathematical, statistical, image processing and other specialised routines (including linear and nonlinear equation solving)
- support for matrix operations
- powerful visualization capabilities, from 2D plots and image displays to interactive 3D graphics that take advantage of OpenGL hardware acceleration
- support for variety of system modelling methodologies within the core *MATLAB* package, as well through optional toolboxes such as *SIMULINK* and *STATEFLOW*
- development of standalone applications facilitated by built-in Graphical User Interface (GUI) builder, *GUIDE*, and optional *MATLAB* Compiler
- *MATLAB Web Server* lets *MATLAB* programmers develop Web-deployable applications. HTML documents serve as a point-and-click GUI for *MATLAB* applications being deployed. Application users are not required to learn *MATLAB* nor need it be not be run locally on client machines.

standalone models: Yes. Models can be built into standalone applications using an extensive set of user-interface tools and the (optional) *MATLAB* Compiler.

open source: *MATLAB* itself is 'open-ended' in that its functionality is extensible and user-customisable, although underlying kernel is proprietary closed-source software. Script codes are distributable in open ASCII text format.

required expertise: Medium - high (research level software with a fairly high threshold of knowledge and steep learning curve for novice user, although user interface is very sophisticated)

platform: Windows and Unix-like (including MacOS-X) and linux systems

cost/licensing: single floating network licence £2995 for commercial users (UK price: December 2005). Add-on toolboxes cost extra. Substantial discounts are available for academic users.

web: The Mathworks : www.mathworks.co.uk

IDL : Research Systems, Inc (RSI)
IDL integrates a high-level programming language with an extensive library of functions and other tools for data analysis, visualization, and cross-platform application development. The feature set is particularly strong with respect to image-based analysis and visualisation and data mining. The software environment caters for interactive analysis and display as well as large-scale commercial programming projects. *IDL* does not include a suite of tools explicitly intended for system simulation, but simulation models can easily be coded, distributed and executed in the form of scripts.

Main features highlighted in marketing:

- high-level language for numerical computation and data analysis
- extensible and customisable through user-defined functions
- a rich library of mathematical, statistical, image processing and other specialised routines
- support for matrix operations
- powerful visualization capabilities, from 2D plots and image displays to interactive 3D graphics designed to take advantage of OpenGL hardware acceleration
- Graphical User Interface (GUI) toolkit for *IDL* 'application' development
- Support for standalone applications distributed in proprietary binary format can be run with the free *IDL Virtual Machine* or with a 'runtime' license of *IDL*

standalone models: Yes, using free runtime-only model 'viewer'.

open source: *IDL* itself is 'open-ended' in that its functionality is extensible and user-customisable, although underlying kernel is proprietary closed-source software. Runtime-only models distributed in closed proprietary format, although script codes can be separately provided in open ASCII text format.

required expertise: Medium - high (research level software with a fairly high threshold of knowledge and steep learning curve for novice user, although user interface is very sophisticated)

platform: Windows and Unix-like (including MacOS-X) and linux systems

cost/licensing: single floating network licence £2995 for commercial users (UK price: December 2005). Substantial discounts available for academic users.

web: RSI – IDL : www.rsinc.com/idl

Mathematica : Wolfram Research, Inc
Mathematica seamlessly integrates a numeric and symbolic computational engine, graphics system, programming language, documentation system, and advanced connectivity to other applications. It is particularly capable in the area of symbolic mathematics and equation solving.

Main features highlighted in marketing:

- powerful and flexible high-level language for symbolic and numerical computation and data analysis
- support for complex symbolic calculations that involve thousands or millions of terms
- support for matrix operations
- large library of mathematical, statistical, and visualisation functions
- powerful visualization capabilities, from 2D plots and image displays to interactive 3D graphics
- ability to embed *Mathematica* functionality into applications developed using high-level languages such as C or Visual Basic, and develop interactive web applications that run on a *webMathematica* server

standalone models: Yes. Models can be built into standalone applications using an extensive set of user-interface tools and the (optional) MATLAB Compiler.

open source: *Mathematica* itself is 'open-ended' in that its functionality is extensible and user-customisable, although underlying kernel is proprietary closed-source software. Script codes are distributable in open ASCII text format.

required expertise: Medium - high (research level software with a fairly high threshold of knowledge and steep learning curve for novice user, although user interface is sophisticated)

platform: Windows and Unix-like (including MacOS-X) and linux systems

cost/licensing: single *Mathematica Professional* licence £1625 for government users (UK price: December 2005). Add-on packages and web-enabled services cost extra. Substantial discounts available for academic users.

web: Wolfram Research : www.wolfram.com

MathCad : Mathsoft
Mathcad is an integrated software environment for performing and communicating maths-related work. Like *Mathematica,* it is particularly capable in the area of symbolic mathematics and equation solving, and as an interactive presentation tool.

Main features highlighted in marketing:

- numeric operators perform summations, products, derivatives, integrals and Boolean operations
- symbolics simplify, differentiate, integrate, and transform expressions algebraically
- vectors and matrices manipulate arrays and perform various linear algebra operations, such as finding eigenvalues and eigenvectors, and looking up values in arrays
- ability to generate random numbers or histograms, fit data to built-in and general functions, interpolate data, and build probability distribution models
- ability to generate random numbers or histograms, fit data to built-in and general functions, interpolate data, and build probability distribution models
- 2D and 3D visualization capabilities
- 'live' maths technology allows use of mathematical expressions using standard mathematical notation - but with the added ability to recalculate, view and publish results easily, including to the Web.
- ability to embed *Mathcad* functionality into applications developed using high-level languages such as C++, Java or Visual Basic, with connectivity with *VisSim*, *MATLAB* and Microsoft *Excel* also featured

standalone models: No, although Mathcad has the ability to interface with external applications such as Microsoft *Excel* and *MATLAB*, as well as applications developed using common high-level programming languages.

open source: *Mathcad* itself is extensible and user-customisable, although underlying kernel is proprietary closed-source software.

required expertise: Medium - high (research level software with a fairly high threshold of knowledge and steep learning curve for novice user, although user interface is sophisticated)

platform: Windows systems

cost/licensing: single *Mathcad* licence £758 for government users (UK price: June 2006). Add-on packages cost extra. Discounts of around 50% or more available for academic users.

web: Adept Scientific : http://www.mathsoft.com/

GNU Octave : John Eaton and others (published via Free Software Foundation)

GNU Octave is freely redistributable software that integrates a high-level interactive language, primarily intended for numerical computations, with 2D and 3D visualisation capability. The software is command line driven, and the command language used is that is mostly compatible with MATLAB (although only a subset of MATLAB functionality is currently implemented). The software does not include tools explicitly intended for simulation, but simulation models can easily be coded, distributed and executed in the form of scripts.

Main features highlighted in marketing:

- high-level (MATLAB compatible) language for numerical computations
- extensible and customisable through user-defined functions
- wide range of elementary functions (including support for Boolean operations)
- specialist functions for linear and nonlinear equation solving
- support for matrix operations

standalone models: No – unlike Matlab or IDL, there is no compiler or suite of embedded tools for application development

open source: Yes (both application software and any user-developed model scripts)

required expertise: High (research level software with a high threshold of knowledge and steep learning curve for novice user)

platform: Windows and Unix-like systems, but requires compilation with freely available GNU C++ compiler

cost/licensing: Free to download and can be freely redistributed under terms of GNU public licence

web: GNU Octave: www.octave.org Free Software Foundation: www.gnu.org

Appendix 3: Geomorphological understanding incorporated into Boolean functions given in Table 3.

The following are not intended as a definitive set of functions suitable for all applications. Rather, they are a first attempt at implementing a Boolean network model in a way that is consistent with the level of abstraction adopted in the formulation of the EstSim behavioural statements and system diagrams. The approach is extensible and further refinements could be implemented to take account of a broader consensus of expert opinion, or even differences in opinion, regarding the functional interaction of the various system components.

External forcing

`OW = ow & ~barrage`

Ocean waves: wave energy can be imposed, but is negated by a barrage.

`LWP = lwp & ow`

Longshore component of wave power; requires ocean waves to be imposed.

`MS = ((ms & ~slr) | (ms & ow)) & ~barrage`

Marine sand supply: if imposed without waves can persist in absence of sea-level rise (relict source); otherwise, requires wave action to mobilise material. Always negated by barrage (and an updrift jetty could have the same effect)

`MM = mm`

Marine mud supply.

`WD = wd`

Wind regime favourable for dune formation (also assumed to favour wave generation over estuarine fetch, although separate parameters could be introduced if required).

`SLR = slr`

Accelerated sea-level rise: existing estuaries assumed to be in equilibrium with present rate of sea-level rise.

`DREDGE = dredge`

Dredging of navigation channel. Currently, a single variable represents outer and inner estuary sub-systems. Effect is to remove material from subtidal (thereby countering any tendency for subtidal infilling).

`BARRAGE = barrage`

Construction of a tidal barrage. Blocks ocean wave propagation, greatly restricts tidal exchange and therfore reduces prism (a small exchange assumed), and blocks marine sediment.

Outer estuary sub-system

`OEW = (ow & ~sp) | (wd & ~tp_o)`

Outer estuary waves. Propagated via mouth in absence of spit and/or generated within estuarine given suitable wind regime and fetch (here modelled simply by a large prism - more sophisticated formulations are clearly possible).

`TP_O = (~osm | ofp) & ~barrage`

Outer estuary spring tidal prism. Reduced (infilled) by saltmarsh and by removal of tidal floodplain or construction of a barrage.

```
TTP_L = ~(tp_o | tp_i)
TTP_M = xor(tp_o, tp_i)
TTP_H = tp_o & tp_i
```

Three state variables used for accounting purposes. Mutually exclusive functions are defined to give a gradation of total estuary spring tidal prism according to a simple truth table formed by the two bit representation of outer and inner estuary prism. These could be eliminated by re-coding the other functions but are retained here to illustrate how gradational responses can be incorporated.

```
DEPTH_RATIO_O = ~((osf_u | omf_u) & osubtidal_fill) | ~((osm | ofp))
AREA_RATIO_O = ~((osf_u | omf_u)  & ofp) | ~osm
```

These two functions attempt to implement, in the simplest way possible, a measure of velocity asymmetry loosely based on Dronkers γ (Dronkers, 1996). This is a first step, and this could probably be refined, especially if an extended (i.e. multi-bit) representation of outer and inner estuary prism were to be incorporated.

```
SG_O = ms & depth_ratio_o & area_ratio_o
```

Bedload (~sand) transport residual based on velocity asymmetry implied by simple Dronkers γ.

```
MG_O = (mm & (osf_u | omf_u | ~tp_o)) | (mm & ~oew)
```

Suspended sediment (~mud) transport residual based on asymmetry implied by intertidal morphology (i.e. tidal flats feedback to increase consolidation and probability of sediment retention, and saltmarsh acts to increase efficiency of intertidal sediment sink.

```
BB = (bb & ~slr) | (ow & ms)
```

Barrier beach. Can be imposed as a relict feature that persists in the absence of sea-level, or formed through constructive action of waves on marine sand supply.

```
SP = (bb & lwp & ms ) | (sp & bb & lwp & ~slr)
```

Spit. Requires an updrift beach, marine sand and a significant longshore wave power. Can persist as inherited feature in absence of new marine sand input provided there is no acceleration in the rate of sea-level rise.

```
DU = (du & ~slr) | ((((bb | sp) & ms) | osf_u) & wd)
```

Dunes. Can persist as relict features in absence of acclerated sea-level rise. Otherwise, require a beach or spit and a supply of sand, or a high sand flat, and a favourable wind regime.

```
ED = ((ttp_l | ttp_m ) & ms & sp & ~orp & ~ic) | (ttp_l & ms & (bb | sp) & ~orp & ~ic)
FD = ((ttp_l | ttp_m ) & ms & sp & ~orp & ~ic) | (ttp_l & ms & (bb | sp) & ~orp & ~ic)
```

Ebb and flood tidal deltas. These form at low to medium tidal prism, given various combinations of sand supply and either an updrift beach or spit. Their formation is inhibited by the presence of a deeply incised outer estuary channel or by the presence of exposed rock platforms.

```
LB = (ttp_h | tp_o) & ms & ~(ed | fd | sp | orp | ic) & ~barrage
```

Linear banks replace tidal deltas at high tidal prism. They are similarly inhibited by the presence of exposed rock platforms or incised outer estuary channels, and by the construction of a barrage (which would reduce the tidal prism).

```
IC = ic
```

Incised estuary channel.

```
OSF_L = (ms & (sg_o & oew)) | (osf_l & ~(~ms & (oew | ~sg_o)))
OSF_U = (ms & oew & osf_l) | (osf_u & ~(~ms & (oew | ~sg_o)))
```

Outer estuary sand flats. Lower flat depends on sand supply and is influenced by velocity asymmetry of inlet (i.e. by tidal transport along the estuary) as well as by waves. Upper flat requires only wave action and requires previous deposition of lower sand flat.

```
OMF_L = (  (mm & ~oew) | (omf_l & ~(~mm & oew))) & ~osf_l
OMF_U = (mm & (mg_o & ~oew) & (omf_l | osf_l)) | (omf_u & ~(~mm & oew))
```

Outer estuary mud flats. Lower flat depends on mud supply and is inhibited by wave action. It is also mutually exclusive with lower sand flat within this portion of the accommodation space. Upper mud flat requires previous deposition of lower sand or mud flat, and is inhibited by waves but favoured by mud residual transport into the estuary.

```
OSM = (omf_u & mm & (~oew & mg_o))
```

Outer estuary saltmarsh. Formation depends on pre-existing upper mud flat, a supply of mud and residual transport of mud into the estuary (which saltmarsh then reinforces), but is inhibited by wave action.

```
OSUBTIDAL_FILL = ((ms & sg_o) | (osubtidal_fill & ~(ofp & ~sg_o)) )
```

```
    & ~(dredge | ri_q)
```

Outer estuary subtidal channel infill. Subtidal deposition favoured by supply of marine sand and landward residual transport (i.e. flood dominance in terms of velocity asymmetry). Once infilled, persists unless sudden prism increase (due to reactivation of tidal floodplain or shift to an ebb dominant regime). Removed by dredging and inhibited by large river discharge.

```
ORP = (orp & ~(osf_u | omf_u | osm)) | ( ~(osf_u | omf_u | osm) & oew &
    (~sg_o | ~mg_o) )
```

Outer estuary rock platform. Usually an inherited resistant feature, persisting in the absence of an intertidal sedimentary cover. Can also form if intertidal sediment cover is removed and estuary has no tendency to infilll again.

```
OFP = ofp;
```

Outer estuary tidal floodplain. By default, a large potential tidal floodplain is rendered inactive by reclamation. This can be reactivated by abandonment or managed realignment. The effect is to increase the tidal prism and alter the flood - ebb dominance of an estuary that has already infilled.

```
OCF = ocf & (oew | ~osm)
```

Outer estuary cliff. An inherited feature, persisting in the presence of significant wave action and the absence of protective saltmarsh.

```
IEW = ((wd | oew) & tp_i) & ~ism
```

Inner estuary waves. Propagated from outer estuary and/or generated within estuarine given suitable wind regime and fetch (here modelled simply by a large prism - more sophisticated formulations are clearly possible). Dissipated by saltmarsh.

```
TP_I = (~(imf_u | isf_u | ism | ifp) & ~barrage)
```

Inner estuary spring tidal prism. Reduced (infilled) by high tidal flat and saltmarsh (inner estuary is assumed to be narrower than outer estuary, such that intertidal deposition has a stronger infleuence on prism) and by removal of tidal floodplain or construction of a barrage.

```
DEPTH_RATIO_I = ~( (ifp | ism) & isubtidal_fill)
AREA_RATIO_I = ~((isf_u | imf_u)  & ifp) | ~ism
```

As in the outer estuary, these two functions attempt to implement, in the simplest way possible, a measure of velocity asymmetry loosely based on Dronkers γ (Dronkers, 1996). Further refinement mght be appropriate, especially if an extended (i.e. multi-bit) representation of outer and inner estuary prism were to be incorporated.

```
SG_I = ms & depth_ratio_i & area_ratio_i & ~ri_q
```

Bedload (~sand) transport residual based on velocity asymmetry implied by simple Dronkers γ.

```
MG_I = (mm & (isf_u | imf_u | ~tp_i)) | (mm & ~iew)
```

Suspended sediment (~mud) transport residual based on asymmetry implied by intertidal morphology (i.e. tidal flats feedback to increase consolidation and probability of sediment retention.

```
ISF_L = (ms & sg_o & (sg_i | iew)) | (iew & ri_s) | (isf_l & ~(~ms & (iew | ~sg_i)))
ISF_U = (ms & sg_o & iew & isf_l) | (isf_u & ~(~ms & (iew | ~sg_i)))
```

Inner estuary sand flats. Lower flat depends on sand supply and is influenced by velocity asymmetry of inlet (i.e. by tidal transport along the inner estuary) as well as by waves. Upper flat requires only sediment input from the outer estuary and wave action, and requires previous deposition of lower sand flat.

```
IMF_L = ( ((mm & (mg_i | ~iew))) | (imf_l & ~(~mm & iew))) & ~isf_l
IMF_U = (mm & (mg_i | ~iew) & (imf_l | isf_l)) | (imf_u & ~(~mm & iew))
```

Inner estuary mud flats. Lower flat depends on mud supply and is inhibited by wave action. It is also mutually exclusive with lower sand flat within this portion of the accommodation space. Upper mud flat requires previous deposition of lower sand or mud flat, and is inhibited by waves but favoured by mud residual transport into the estuary.

```
ISM = (((imf_u | isf_u) & mm & ~iew) | (ifp & mm & ~iew)) & ~(slr & ~ifp)
```

Inner estuary saltmarsh. Formation depends on pre-existing upper tidal flat and a supply of mud, but is inhibited by wave action. Can also form when potential tidal floodplain reactivated, provided there is a supply of mud and negligible wave action. However, will be squeezed out of existence by accelerated sea-level rise unless new accommodation space is provided (by reactivation of potential tidal floodplain).

```
ISUBTIDAL_FILL = ((ms & sg_i) | (mm & mg_i) | (isubtidal_fill & ~(ifp & (~sg_i | ~mg_i)))
)  & ~(dredge | ri_q)
```

Inner estuary subtidal channel infill. Subtidal deposition favoured by supply of marine sand or mud and landward residual transport. Once infilled, persists unless sudden prism increase (due to reactivation of tidal floodplain or shift to an ebb dominant regime). Removed by dredging and inhibited by large river discharge.

```
IRP = (irp & ~(isf_u | imf_u | ism)) | ( ~(isf_u | imf_u | ism) & iew &
       (~sg_i | ~mg_i | ri_q) )
```

Inner estuary rock platform. Usually an inherited resistant feature, persisting in the absence of an intertidal sedimentary cover. Can also form if intertidal sediment cover is removed and estuary has no tendency to infilll again.

```
IFP = ifp
```

Inner estuary tidal floodplain. By default, a large potential tidal floodplain is rendered inactive by reclamation. This can be reactivated by abandonment or managed realignment. The effect is to increase the tidal prism and alter the flood - ebb dominance of an estuary that has already infilled.

```
ICF = icf & (iew | ~ism)
```

Inner estuary cliff. An inherited feature, persisting in the presence of significant wave action and the absence of protective saltmarsh.

```
RI_Q = ri_q
RI_M = ri_m
RI_S = ri_s
```

River inflow (significant water, mud and sand discharge).

Appendix 4: Main morphological components for EstSim generic estuary types as defined in FD2117 Project Report 2 '*EstSim Behavioural Statements Report* 2' (EstSim Consortium, 2004). A simpler variable set is used at this level of abstraction.

Morphological variables

Barrier beach	BB
Sand spit	SP
Ebb and flood deltas	ED, FD
Linear banks	LB
Channels	CC
Sand flat	SF
Mud flat	MF
Saltmarsh	SM
Dunes	DU
Cliff	CF
Rock platform	RP
River	RI

1. Fjord
Morphological variables (potential): BB, SP, CC, SF, CF, RP,RI

2. Fjard
Morphological variables (potential): BB, SP, CC, SF, MF, SM, CF, RP,RI

3. Ria
Morphological variables (potential): BB, SP, CC, SF, MF, SM, CF, RI

4. Spit –enclosed
Morphological variables (potential): BB, SP, ED, FD, CC, SF, MF, SM, DU, CF, RI

5. Funnel-shaped
Morphological variables (potential): BB, LB, CC, SF, MF, SM, DU, CF, RI

6. Embayment
Morphological variables (potential): BB, LB, CC, SF, MF, SM, DU

7. Tidal inlet
Morphological variables (potential): BB, SP, ED, FD, CC, SF, MF, SM, DU, CF

Appendix 5: Example of steering file format for prototype simulator

```
%1 Steering file for ESTSIM boolean system simulator
%2 Run:   ESTSIM WP5.2 Ribble case study : 1951 input
%3 Notes: V10 Boolean variable set and function library
%4 Notes:
%5 Format: first line contains number of variables, subsequent lines variable names,
%6 Format: followed (optionally) by number of specified initial states and specified states
41
ow
ms
mm
wd
slr
dredge
oew
tp_o
ttp_l
ttp_m
ttp_h
depth_ratio_o
area_ratio_o
sg_o
mg_o
bb
sp
du
ed
fd
lb
osf_l
osf_u
omf_l
omf_u
osm
osubtidal_fill
ofp
iew
tp_i
depth_ratio_i
area_ratio_i
sg_i
mg_i
isf_l
isf_u
imf_l
imf_u
ism
isubtidal_fill
ifp
1
1  ow
1  ms
1  mm
1  wd
0  slr
0  dredge
1  oew
1  tp_o
0  ttp_l
1  ttp_m
0  ttp_h
1  depth_ratio_o
1  area_ratio_o
1  sg_o
1  mg_o
1  bb
0  sp
1  du
0  ed
0  fd
1  lb
1  osf_l
1  osf_u
0  omf_l
0  omf_u
```

```
0  osm
0  osubtidal_fill
0  ofp
0  iew
0  tp_i
1  depth_ratio_i
1  area_ratio_i
1  sg_i
1  mg_i
1  isf_l
0  isf_u
0  imf_l
1  imf_u
1  ism
0  isubtidal_fill
0  ifp
```