

Research Article

Implementing a new data model for simulating processes

F. REITSMA*† and J. ALBRECHT‡

†Institute of Geography, School of Geosciences, The University of Edinburgh,
Edinburgh EH8 9XP, UK

‡Department of Geography, Hunter College, CUNY, USA

(Received 26 January 2005; in final form 27 April 2005)

The paper describes the development of a new methodological approach for simulating geographic processes through the development of a data model that represents a process. This methodology complements existing approaches to dynamic modelling, which focus on the states of the system at each time step, by storing and representing the processes that are implicit in the model. The data model, called *nen*, focuses existing modelling approaches on representing and storing process information, which provides advantages for querying and analysing processes. The flux simulation framework was created utilizing the *nen* data model to represent processes. This simulator includes basic classes for developing a domain specific simulation and a set of query tools for inquiring after the results of a simulation. The methodology is prototyped with a watershed runoff simulation.

Keywords: Process; Modelling; Simulation; Data model; GIScience

1. Introduction

There are many different methodologies for modelling geographic processes, such as partial differential equations or agent-based modelling (ABM). Any of these approaches assume a certain conceptualization of the entities they are concerned with, whether it is explicitly formalized within an ontology or implicit in the underlying assumptions of the model. This paper presents a new data model for simulating processes that aims to advance process modelling. The approach is founded on a theory and subsequent conceptualization that takes *process* as the modelling primitive. The advantages of raising process to the fore lie in the ability to pose novel types of questions and explore process dynamics and their causal interactions.

This process perspective contrasts with current approaches to modelling processes, where the process, while formalized in the model, is not explicitly represented. Rather, the state of the modelled system at each instant of time is typically represented. As such, the methodology presented in this paper provides a complementary technique to traditional approaches. Section 2 explores the representations used in modelling geographic processes by considering current methods in the light of their conceptual underpinnings. Section 3 follows with a description of a process-oriented data model, which forms the basis for querying and analysis of processes. The structure of the simulator that implements this data

*Corresponding author. Email: femke.reitsma@ed.ac.uk

model is then given in section 4, and the results of a small test case implementation are presented in section 5. Section 6 concludes the paper.

But first, some caveats. In what follows, reference to an *object* in terms of object-oriented implementation will be clearly stated in order to avoid confusion with the use of the term object to represent a static primitive. Furthermore, unless otherwise specified, the use of the term *process* in the paper will typically refer to a geographic process such as erosion, sediment deposition, or migration, as opposed to a computational process.

2. Modelling geographic things

As expressed in the introduction, it is assumed that geographic process models do just that, model geographic processes. However, it is argued here that this is precisely not what typical modelling methods do. In what follows, four arguments are presented for a methodology that takes process as its primitive; namely, it is processes which should be modelled rather than future system states, the need for storage and query of process information, the potential for process analysis and uncovering causality within models, and the utility of the process construct as the basis for interoperability and greater query and analysis efficiency. These arguments are not predicated on what cannot be done, rather, on what is not being done in dominant approaches to modelling geographic processes due to the focus on modelling future system states.

2.1 Modelling processes

Every knowledge base or knowledge-based system is committed to some conceptualization, either explicitly or implicitly (Gruber 1993). Similarly, modelling methods are also constrained by an explicit, or more commonly, implicit conceptualization. Typical approaches to modelling geographic processes are committed to conceptualizations that focus on modelling future system states rather than the processes themselves. Between state time slices, amendment vectors (Langran 1993, Peuquet 1994, Wachowicz 1999), cellular automata (CA) state changes, and agent movements (Benenson and Torrens 2004), the nature of the process is not explicitly represented and recorded. While processes are specified as rules or equations in traditional approaches, there are no data models or data structures that represent process dynamics, regardless of whether they can be derived by reevaluating the rules between time slices. As expressed by Claramunt *et al.* “[c]urrent spatio-temporal models are oriented toward the representation of the evolution of spatial entities. However, none of them provides basic constructs to specify the underlying knowledge describing processes occurring in the real-world” (Claramunt *et al.* 1997, p. 423).

GIS are committed to an implicit conceptualization based on static objects or system states, where temporal representations are mainly concerned with the states and changes of states of these objects or fields (Yuan 1996). The typical data model primitives available to the user are points, lines, polygons, and pixels, or combinations of these (Cova and Goodchild 2002). As a consequence, temporal extensions to GIS are lacking in their ability to reason about and model processes (Clarke *et al.* 2001, Frank 2001, Pang and Shi 2002, Raper and Livingstone 1995, Worboys 2001). These inadequacies of current GIS to support processes are due to a lack in theoretical foundation (Kavouras 2001).

From their earliest days GIS were not designed or pre-conceptualized as dynamic modelling tools. However, they have been extended to represent dynamic phenomena; the two main approaches being: temporally extending GIS, and coupling GIS to environmental models. Temporal extensions to GIS typically involve either snapshots, where each layer represents an instance in time, or amendments vectors, where each entity is associated with a list that contains information regarding each change in the entity (Langran 1993, Peuquet 1994, Wachowicz 1999). For both these approaches, change is interpolated between consecutive system states, whether it be between system states or object states. Alternatively, time can be represented by space, as has been developed in time geography which implements Hägerstrand's classic model of temporal phenomena (Hägerstrand 1967, Miller 2004). Computational implementations of time geography represent the potential path of an individual as a spatial extent which changes over time as the individual moves through space over time (Bernard and Kruger 2000).

Similar difficulties are found in modelling approaches linking GIS and dynamic models, even with the purported saviour of integration and process representation, object-orientation (Bian 2000, Raper and Livingstone 1995, Wachowicz 1999). The development of object-orientated programming languages has engendered much research in object-oriented GIS, modelling, and databases. However, object-orientated approaches typically handle time by time-stamping objects or their attributes (Stefanakis 2003), where change is represented as the difference between an old state and a new state with a new time stamp (for example Yuan 1996, Yuan 2001, Zhang and Hunter 2000), with no reference to the processes that might have changed those objects in the model.

The data models of CA and ABM, although dynamic, are still based on system or object states at instants of time; “[e]ach agent has internal states and behavioural rules. Some states are fixed for the agent’s life, while others change through interaction with other agents or with the external environment” (Epstein and Axtell 1996, p. 4). Modelled processes are typically represented as the relationship between the current and future states of cells or agents, defined by a set of behavioural rules. Processes are therefore implicit to the model, embedded in the rules of the agent or cell, yet they are not explicitly modelled, nor can they be directly inferred from changes between recorded system states. For example, in an ABM of urban sprawl, each agent may have a set of behavioural rules defining their movement and interactions. At each time step, the system state is logged in the form of agents and their attributes. However, whether the future system state of sprawled urban form is a direct result of processes such as rent increases in the inner city or increases in crime, is not represented or stored. The extent of an ABM’s ability to discuss process is to link the initial model setup or specification with the output through some form of spatial pattern metric, where the measure of spatial pattern provides some indication of which processes occurred where (Parker and Meretsky 2004, Rand *et al.* 2003).

Similarly, equation-based models (EBMs) also focus on system states and their update. An EBM, in its simplest form, is a function that can be applied to some observable, and in its spatial form, is typically a partial differential equation. These observables are measurable characteristics of interest that may change over time. EBMs are based on a set of equations that express relationships among observables, their evaluation producing the evolution of the observables over space and time. The

equation itself represents the process, but its operation is not recorded. As with ABM, in EBM there are ad hoc solutions for determining the path of a process and which process is operating where, but no general solution or data model which addresses this directly. The vector field represents both direction and magnitude at each instant of time; for example, wind fields. This comes much closer to the data model represented here. However, vector fields are utilized to represent the movement of some mass as opposed to the processes that are involved in that movement.

The modelling methodology presented in this paper focuses on the representation and storage of processes expressed in current models with a process-oriented data model, complementing existing methods of process modelling. This approach avoids the loss of information through the cracks of time, such as through the imposition of an inappropriate temporal granularity that misses changes, as it requires representation at the level of the defined process.

2.2 Storing and querying processes

Within the field of spatio-temporal Database Management Systems (DBMS), spatial formalisms have been temporally extended (Abraham and Roddick 1999, Griffiths *et al.* 2001). Traditionally spatio-temporal DBMS involved extensions of the relational data model (Peuquet 2001), yet of late there has been a transition from relational data models to object models (Griffiths *et al.* 2001). The focus of spatio-temporal data modelling for spatio-temporal DBMS is on objects and their relationships, such as their spatio-temporally extended entity-relation model (STER) (Huang and Claramunt 2002). These objects and relationships are temporally extended and have histories that specify their changes, where the object or the attribute is time-stamped. For example, MOD (Moving Objects Database) systems are designed for applications such as tracking delivery vans, taxicabs, or military vehicles (Libourel 2001). As discussed earlier with modelling approaches, this focus on representing objects at an instant of time results in a loss of information about processes.

In terms of change, there are two types typically evident in a database: schema evolution and data evolution (Erwig *et al.* 1999). For data evolution, most spatio-temporal database modelling emphasizes the snapshot view, where change can be interpolated between time slices of system states or object states (Claramunt and Parent 2003). These changes have also been used in constraining the evolution of objects represented in a database, defining permissible and prohibited evolutions in the database where evolution or change is modelled as a temporal relationship between two states. More recently Mountrakis *et al.* (2002) developed a change-oriented data model for the storage and querying of spatio-temporal information, which allows them to store the change between time slices that represent objects such as buildings or cadastres, and query those changes at multiple levels of abstraction. However, in order to understand the changes in our modelled system we need to know the processes that caused those changes, that is, to explicitly store causal relations defined in our model.

Storing the process information of a dynamic model allows for process queries. Data can be mined for process information or classified into process types automatically or manually (Merz and Blöschl 2003, Yuan 2001), however current approaches to representing model results do not allow for easy querying of process information. For example, figure 1 expresses this difficulty. Here the location of the

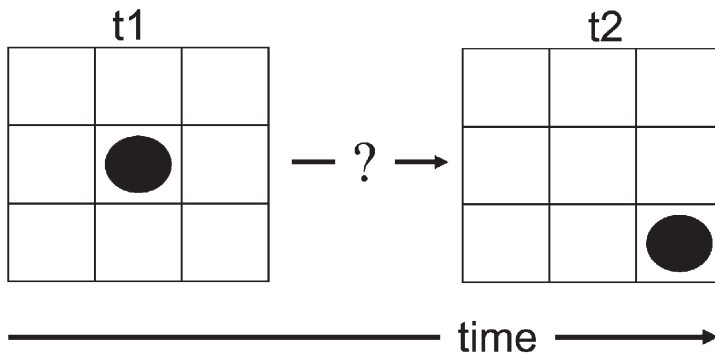


Figure 1. Process inference.

black point moves from time one (t1) to time two (t2), yet given knowledge of the system state at each of those times, the process by which the point moves is not stored. Our ability to determine the process typically depends upon an in-depth knowledge of the model and the system it represents, and has the potential to result in the wrong process. In order to accurately determine the processes in operation the model must be rerun, applying the rules or equations over again. However, there are currently no common data models for representing processes, therefore extraction of this process information leaves us with no way to analyse or query it.

In the traditional theoretical framework, by definition process is something that occurs between system states. That is, process is the translation between system or object states at different times, therefore it cannot be represented in one time slice. Consequently, queries about where a process is occurring at an instant of time cannot be expressed with current approaches. Only two basic types of queries may be asked of attributes of the representation: “what is at a specific location?” or “where is a certain attribute?”, the composition of which define the realm of possibilities (Goodchild 2003, Peuquet 2002). With the dynamic extensions of ABM, CA, and EBM, these queries are temporally qualified, yet there remain the two fundamental types of queries that can be asked. For example, given a specific agent, what are its associated properties at time x ? Or, given a specific set of cells (i.e. location), what are its associated properties at time x ? In terms of change queries, attributes and entities are queried as to if and when they changed by interpolating between these states.

Spatio-temporal databases are designed to store historical, present, and possible future data (e.g. for planning purposes), “they are not designed to record which processes activate a change” (Claramunt *et al.* 1997). To understand, query, and explain processes, processes must be represented and consequently stored. How or why questions cannot be easily asked or answered with a computational method based on current approaches focused on what, where, and when questions.

2.3 Process analysis and causality

Modelling a process is not merely tracking and storing the movement of some object, such as an agent. Recording change does not equal process. For example, recording the change of landscape morphology does not give an indication of the processes causing its change, such as erosion or tectonic uplift. Clearly change in the attributes of entities can be recorded and associated with changes in model structure

or initial conditions. But with current data models we cannot hunt the processes that caused those changes.

Analysing the interaction of processes is important to determining how various processes propagate through the system over time, and to ascertain which spatio-temporal points in the model to tweak. In simulating processes, insights may be gained into their causal relations by storing information about their interactions. Questions regarding how the rules of the process affect the dynamics of the process (rather than the pattern produced by the process) may be better explored by modelling and storing process information.

2.4 Efficiency and interoperability

In querying or analysing processes, an argument can be made for the inefficiency of attempting to recreate processes each time in order to query the results as to where certain processes caused changes in system states. The proposed methodology of explicitly storing process information attempts to overcome this problem, allowing for queries similar in nature to current system state queries, such as querying for the location of processes, their attributes, interactions, or their change over time. Furthermore, state information can be derived from the modelling approach presented in this paper, so there is no loss of information. For example, in modelling the process of coastal erosion, the various eroded states of the system can be directly extracted from the process model, as will become clear in the methodology discussion below. However, storing this added level of information also results in a new explosion of information, akin to the overload currently being experienced with system state data such as remote sensing imagery.

The proposed approach of modelling and simulation with process as the single primitive provides a basic construct, which if applied to models of different domains could facilitate interoperability between models. Common representations of space-time, which has been one of the key problems of integrating GIS and environmental models, potentially allow interoperation at the process level rather than the model level, removing the effort required in translating between models. This could be an important boon to modellers of complex systems deriving their model components from different fields of study and the future development of eScience modelling initiatives on the Grid (Pouchard *et al.* 2003, Reitsma and Albrecht 2005).

3. Process data model

A process data model is the single modelling primitive used in the simulation framework discussed in section 4 below. This representative device can be expressed in tuple form as:

$$(x1, y1, x2, y2, st, \{a1, a2, \dots\}, \{r1, r2, \dots\})$$

Or it may be presented graphically as a (node,edge,node) triple, as illustrated in figure 2. Each (node,edge,node) will be henceforth referred to as a *nen*. The location of the process is identified by $x1, y1, x2, y2$, which expresses the spatial extent of the process. The st represents the spatio-temporal granularity of the process, which may be a function of the amount of energy that initiates the process. For example, given some threshold breaking push, the spatio-temporal granularity expresses how far and over what time period the process will operate in response to that push. The set $\{a1, a2, \dots\}$ defines the set of attributes of the process. The set $\{r1, r2, \dots\}$ defines

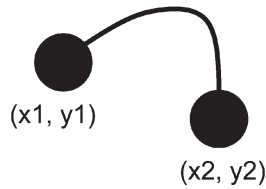


Figure 2. Process representation.

the set of rules of the process that govern its dynamics and interaction with other processes. For example, a set of rules for modelling the process of sediment transport in the longshore may define the spatio-temporal extent of an instance of that process as 5 m/h, depending on various relationships it holds between other processes operating in the nearshore.

Note that this is only a representation of a point process, which might best represent processes such as runoff in a watershed. It can also be extended to areal or linear features and into the third spatial dimension, representing processes such as sediment transport in the nearshore, migration, and El Niño.

4. Simulator structure

The implementation of the conceptual model lies in a field of possibilities. Varying the approach taken to implement a conceptual model, although a technical issue, will also have implications for the results of the model (Gulyás 2002). While recognizing this conundrum, one must begin somewhere. In what follows the approach taken will be described, including some of the design issues and assumptions in the development of the process simulation tool, which is called *flux*. The simulator presented is but one implementation of the general concept of representing process and of the data model discussed above.

From the discrete confines of the computer to the imposed structure of object-orientation, technologically the model is constrained to a particular framework. The straitjacket of choice is Java, including the incorporation of the RePast (Recursive Porous Agent Simulation Toolkit) library, an open source agent-based modelling environment created by Social Science Research Computing at the University of Chicago (<http://repast.sourceforge.net/>). RePast is primarily used for its display and scheduling classes, and also has the advantage of containing Java classes for importing GIS raster data (ASCII raster files). As a caveat, the agent-based environment is not used to do agent-based modelling per se; rather, its classes are used in order to simulate process as the primitive modelling construct.

4.1 Parameterization

For the sake of modelling, a new basic construct is introduced that extends the ontology from the single primitive of process, to a type of restricted process, termed here a parameter. Parameters are instituted due to the difficulty of defining a complete system of processes in any domain, or indeed, modelling the whole world, and typically represent the external input to the model. A process can be modelled as a parameter in the sense that it is an encapsulated process, where none of the internal workings of the process are evident in the parameter, merely a representative value.

Parameters are practical abstractions for modelling geographic processes that are purposefully defined by the researcher in two scenarios. First, parameters are defined when we do not want to or cannot model the whole process, for reasons such as minimizing the complexity of the processes modelled or restrictions imposed by software, hardware, or other external influences. Second, parameters are defined when the observed temporal grain of the phenomenon exceeds the temporal extent of the model. For example, in the first case, to model the process of runoff in a watershed the process of precipitation must be included; however, we may not want to model the whole process of precipitation. Precipitation can then be included in the model as a parameter, represented as a value at a point or over some area to be used by the runoff process model.

Extending this example to the second case, the geomorphology of the watershed may be considered a parameter in the runoff model. Changes in geomorphology are measured with a temporal granularity that exceeds the temporal extent of the process model, that is, geomorphologic changes are observed to take longer than the time the model takes to run, yet they are included because geomorphology has an impact on runoff processes.

Parameters impact on the processes being modelled and can be modified by those processes. However, they have no behaviour of their own. Parameters influence processes whereby the process registers its presence and value at a specific location. Parameters are modified by processes when their values are changed by a process. For example, in a model of erosion, the erosion process will affect the geomorphology, and the geomorphology will influence the dynamics of the erosion process. Yet, geomorphologic change is outside the temporal extent of the model and therefore geomorphology has no defined behaviour of its own.

4.2 Flux

The simulator, called flux, inherits and extends a number of basic operating classes from RePast, namely scheduling classes, display classes, and a base model class. The objects developed in the flux package in turn form the base set of classes for a domain model (figure 3). The flux package contains a set of interfaces and default classes that define the basic structure of the process model, including methods that must be implemented by an inheriting domain model. The objective was to develop as much generic functionality within the flux classes, thereby minimizing the code to be developed within the domain model.

The process model consists of three base classes from which domain-specific models may inherit methods and properties, namely: process, parameter, and model. The model class forms the modelling environment for the processes and parameters; it is incorporated in order to define operational aspects such as the initiation of the model, its display, and parameter scheduling. The process and parameter classes define the common properties and methods that all inheriting process and parameter instances implement. All aspects of the model are conceptually encapsulated within

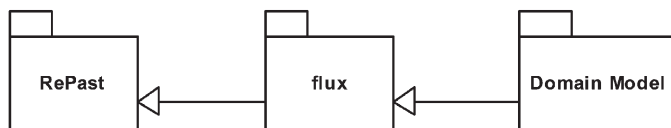


Figure 3. Model inheritance structure.

these three classes. The model class only contains methods pertaining to the setup, scheduling, and recording of the processes and parameters. The setup method creates the processes and parameters that initiate the model. The scheduling method iterates over the parameters and specifies the creation of the process instances based on the thresholds defined in the methods of each process class.

The general class structure of the modelling primitives in the flux package is presented in figure 4 below; a modified Unified Modelling Language (UML) class diagram. The `STEntity` is the top-level interface that specifies the methods that any inheriting process or parameter instance, such as `ProcessDefault` and `ParameterRasterDefault`, must implement. For example, these methods include set and get methods for the properties: temporal grain, spatial grain, temporal extent, and spatial extent.

The `Process` interface extends the `STEntity` interface with added methods that an inheriting process is required to implement. For example, set and get methods for properties defining the location of the process, that is, the `x1`, `y1`, `z1`, `x2`, `y2`, and `z2`. The `ProcessDefault` class implements the `Process` interface with a set of generic properties and methods that are widely applicable to processes in other domains. For example, methods that take care of the display of the process as a node-edge-node triple and the recording of the process are included in this interface. Inheriting classes would then specify methods defining their own behaviour and for the creation and destruction of other processes as a consequence of interactions.

The `Parameter` interface specifies various get and set methods for a parameter, such as its `ID` and `Value`. The `ParameterRasterDefault` is but one implementation of `Parameter`, and extends `RePast's RasterSpace` class to incorporate added functionality such as a generic method for raster colouring. In contrast to a process class, a parameter class is not spatially dynamic, that is, it does not have a changing set of `x1`, `x2`, or `y1`, `y2` properties. Rather, it is located at a point or over an area. This conforms to the classic data models of point, line, polygon, and pixel. The parameter contains the following properties: `temporalGrain`, `temporalExtent`, `spatialGrain`, `spatialExtent`, and `inputFile`. The `temporalGrain` of the parameter defines how often it is updated; for example, precipitation as a parameter may be updated hourly with new input. The `temporalExtent` defines the total number of times the parameter is updated. The `spatialGrain` and `extent`, although typically

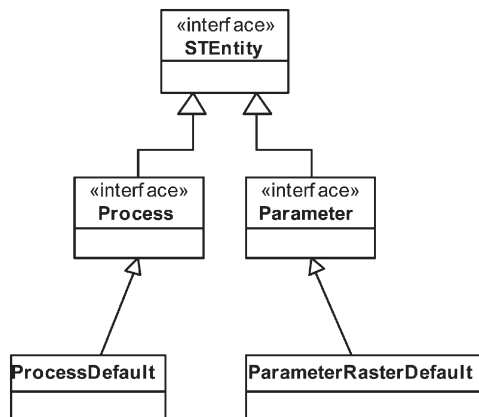


Figure 4. Model class structure of primitives.

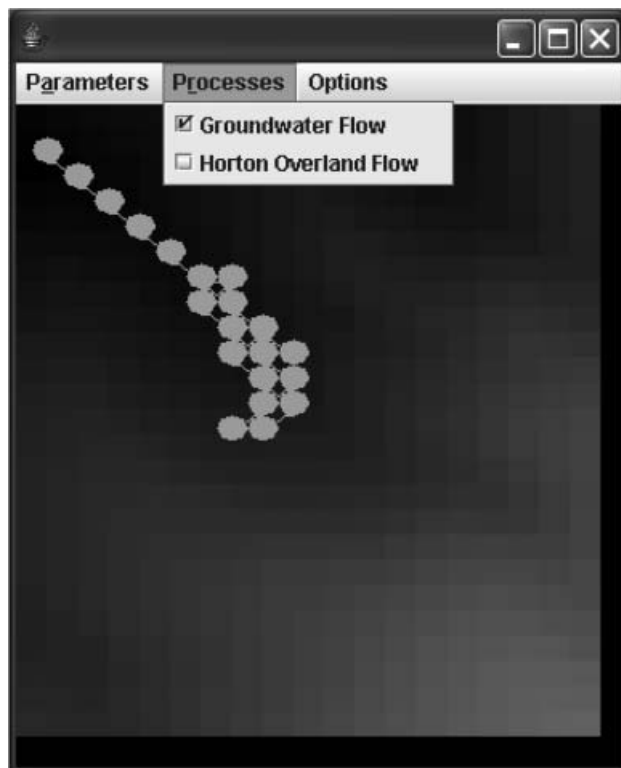


Figure 5. Sample flux display.

implicit in the input file of a raster or vector layer, is specified as it may form the basis of the spatial extent of the process.

Figure 5 above presents the display of a sample simulation. The visible raster layer is a digital elevation model, and the process is groundwater flow.

4.3 Behaviour

For modelling processes, three notions of space-time are subscribed to: absolute, relative, and relational. In absolute space-time the four axes of space-time are used as a measurement framework, describing the relationships among processes through time, dictating the update of input parameters, and initiating the model. Within this absolute spatio-temporal reference framework, processes create a relative space-time through their behavioural rules and properties. This internal time relative to processes' internal dynamics, defines their temporal extent with reference to the absolute framework. Thirdly, each process experiences relational space-time when other processes or parameters influence it. For example, the relative space-time of a process could change in response to synergistic forces with other processes.

In creating this spatio-temporal manifold, the behaviour of a process is defined by a set of rules. These rules not only define the dynamics of each process in relation to parameters, but the interaction among processes. Whenever a process changes, it records its identity and properties to an external database, which forms the basis for query and analysis.

4.4 Queries

The output of the process model is used to query processes for their state at an instant of time or their dynamics over an interval of time. This is in contrast to typical approaches to modelling, which allow for queries of the state of the system rather than the processes that caused that state. These two base types of queries can be applied to properties or attributes of the processes, which includes spatial location. Given the nature of the process data model, the spatial character of a process includes: direction, location, and extent. The results of a simulation are queried with SQL by utilizing the JDBC API to access and query the database via an ODBC interface to connect to the database (<http://java.sun.com>). Depending on the type of query, the output can be provided in text file, graph, or visual display (figure 6).

A GUI has been developed to simplify querying of the database, allowing the user to query for states and changes of the processes stored in the database. These two types of queries extend the system state queries by resolving the processes that are occurring at each instant of time or over an interval of time. Furthermore, system states can be determined from the attributes of the processes.

4.4.1 Process state query. Process state queries characterize the state of the modelled system at an instance or over an interval of time. For example, questions such as “Where is a process over an interval of time?” or “What process is operating at an instant of time?” can be asked based on the process’ attributes or spatial characteristics. For spatial queries, this can include the location and the direction of the process at an instant of time.

The results of process state queries at an instant in time or over an interval of time can be represented as a table of process instances or represented visually as a static display of the processes within the space defined by the model, for example, the distribution of infiltration processes within the space defined by a watershed

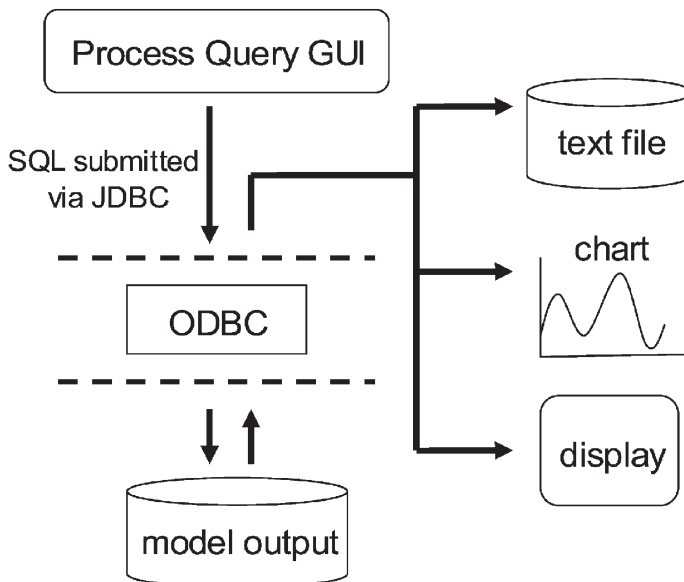


Figure 6. Query operation over database.

parameter. Additionally, in the case of a query over an interval of time, a graph can be produced that represents some attribute or a count of the selected processes (y-axis) over the interval of time (x-axis). The values of location may also be specified as particular values of X and Y or with any other integer operators.

4.4.2 Process change query. A process change query involves the search for patterns of change that define the dynamics of the process over an interval of time. As with process state queries, the three outputs of table, display, and graph, also apply to process change queries. The attribute change of a process over an interval of time can be queried in a number of qualitatively different ways. For example, find processes that have changed an attribute:

- from value a to value b
- from positive values to negative values
- from greater than a to less than b
- from the range a to b to the range c to d
- by percentage or absolute change

More complicated expressions can then be built up from these simple primitives, defining complex patterns of change.

The spatial change of a process is based on the location attributes of the process: x_1, y_1, x_2, y_2 . With the *nen* data model, the basic form of query is defined as a change in location; change in orientation is also included as it is a useful qualitative abstraction that has meaning in models of processes where direction is important. The change of location of a process can either be defined with a specific (x_1, y_1, x_2, y_2) location or with a region, such as that defined by a bounding box. Thus there are four basic combinations: from specific location to location, from specific location to region, from region to specific location, or from region to region. For example, in figure 7 below, a query can be expressed that searches for processes that moved from the dashed square at time one (t_1) to the dashed square at time two (t_2).

For orientation, the query involves specifying the change in the relationship between the x_1 and x_2 and/or y_1 and y_2 . The relationships are specified by the three relational operators: equals ($=$), greater than ($>$), and less than ($<$). For example, figure 8 illustrates the following query: select processes that have changed in orientation such that the process attribute $y_{2,t_2} > y_{1,t_1}$.

Beyond the simple process query, which is a basic analytic device, new quantitative measures need to be derived from the process model that allows for

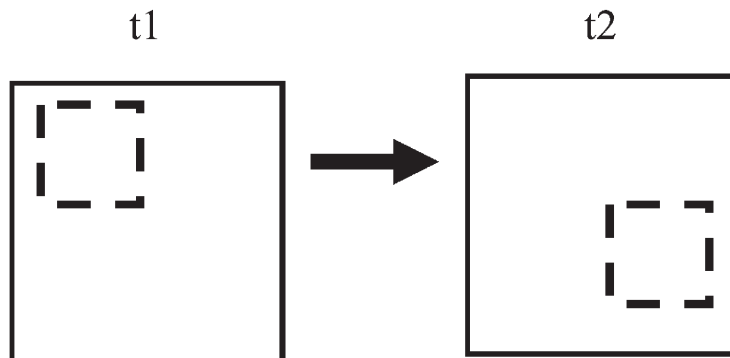


Figure 7. Example of a spatial change query.

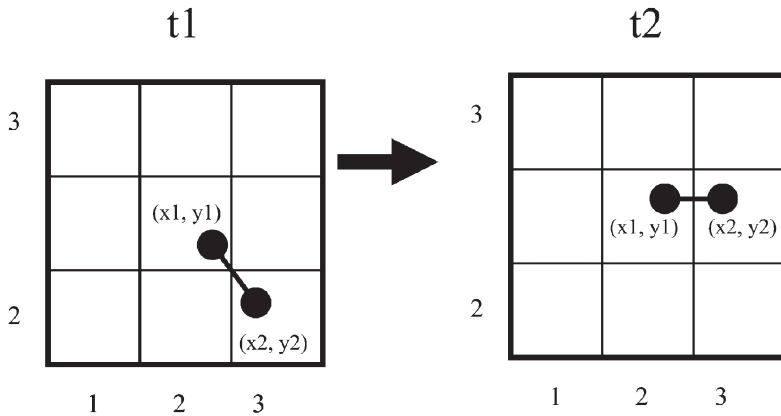


Figure 8. Example of a process spatial change query for orientation.

comparison between models. This and other analytical questions go beyond the scope of this paper, but form the obvious next step towards a better understanding of the operation of processes.

5. Simulation and results

The results of the queries may be displayed in a chart, two-dimensional display, or text file, depending on the query type. For example, displaying results in a chart only applies to queries for a certain quantity over time, such as the value of an attribute from time step 5 to time step 45. A sample chart output is displayed below in figure 9, where time is the x-axis, and a count of processes from a dummy simulation is the y-axis. The chart display utilizes the JFree Java library, which includes classes for plotting charts.

In order to simulate the model, it was necessary to introduce two new classes: ProcessController and ParameterController. These two classes were implemented in order to control their respective process and parameter classes and instances, providing a useful intermediary between the process model and the process classes. These two classes are defined in the flux package, where the ParameterController is

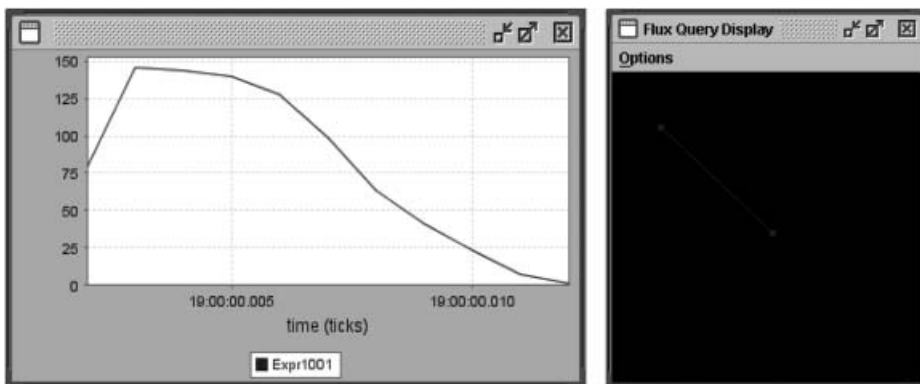


Figure 9. Sample chart, and graph output.

an interface with methods to be implemented, and the ProcessController forms an abstract class with a few generic methods.

5.1 Simulation behaviour

A sample operation of the model is depicted in Figure 10 below as a UML activity diagram. At the initiation of the model a series of setup methods are implemented, such as the creation of the ProcessController and ParameterController and the display surface. The model then iterates over a set of commands that update any of the parameters needing to be updated, calls the ProcessController to operate its processes, updates the display, and then calls a method that records the results of each process in a text file at the end of the model run. When the Process controller is called to operate, it iterates through each process until the process runs out of energy. This property of process energy is used to calibrate the relative and relational spatio-temporal extents of the process with the parameter defined model update. Each time a process instance is created or changed it is recorded in a text file containing all records of the class of processes it belongs to. Currently the ID, location, energy, and value of the process are recorded. However, this can be extended to any property of the process.

As expressed earlier, the scheduled time forms the absolute framework within which relative and relational notions of time are implemented. The scheduled time is typically defined by an input parameter, such as the hourly input of precipitation; the relative time of associated processes is specified by the operation of the process; and the relational time is defined by its interaction with other processes. Each

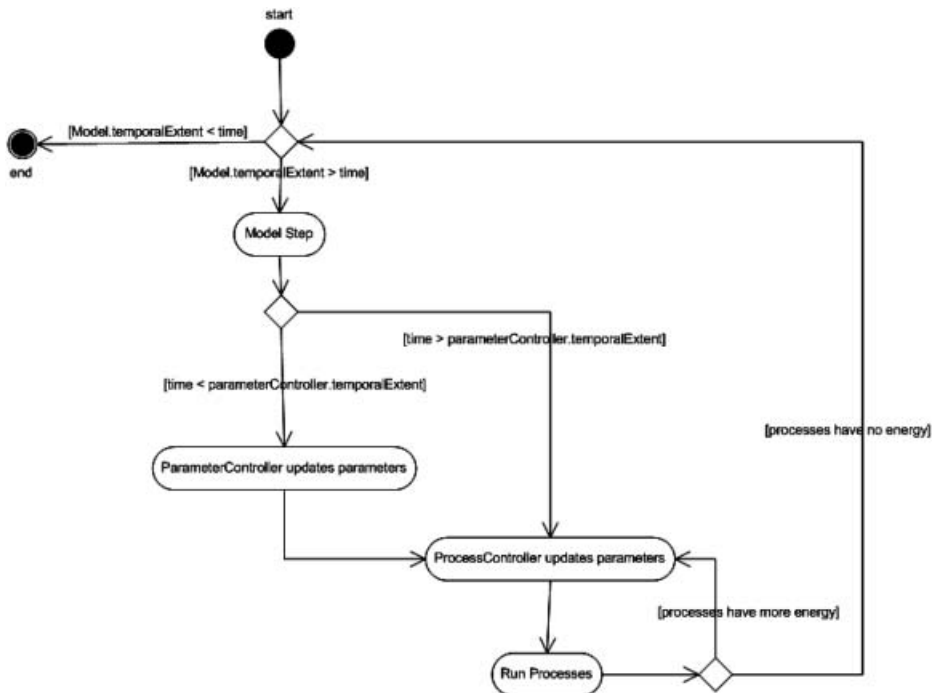


Figure 10. Sample simulation diagram.

operation or interaction requires a certain amount of energy, which is relative to the absolute time defined by the scheduler.

5.2 Sample simulation

For the purposes of testing the methodology a very simple watershed model was simulated, which describes the hydrological processes of the watershed. The watershed model involved the following restricted set of processes: Hortonian overland flow, groundwater flow, infiltration, percolation, saturation excess runoff, and surface ponding. The data used to define the parameters for the simulation are taken from the Reynolds Creek Experimental Watershed (RCEW), which is a high-quality long-term dataset created by the US Department of Agriculture Agricultural Research Service's Northwest Watershed Research Center in Boise, Idaho, United States (<http://www.nwrc.ars.usda.gov>). For a full description of the RCEW, see the special issue of *Water Resources Research* introduced by Marks (2001).

A subset of the RCEW was selected in order to develop and test the simulation, namely Upper Sheep Creek. The input parameters clipped to the bounding box describing Upper Sheep Creek include a digital elevation model, an infiltration capacity layer, a layer defining hydraulic conductivity, and hourly updated data layers of precipitation. The infiltration capacity was derived from soil data, utilizing the soil hydrologic group. The hourly precipitation data layers were generated by interpolating a surface over the whole watershed before clipping these layers to the Upper Sheep Creek subset.

At each hourly time step the precipitation input is updated, which initiates one of three processes, Hortonian overland flow, infiltration, or surface ponding. Each process type has a set of rules defining its behaviour. For example, the rule defining the initiation of these processes expresses that if the precipitation exceeds the infiltration capacity of the soil and depending on the slope characteristics, an instance of Hortonian overland flow will be generated; otherwise either infiltration will occur or if there is no downhill slope and the precipitation exceeds the infiltration capacity surface ponding will occur. The spatio-temporal dynamics of the groundwater flow, Hortonian overland flow, and saturation excess flow is governed by the simple D8 rule that routes the process in a single direction based on the minimum elevation in its eight cell Moore neighbourhood. Although hydrologically limited, the example presents the advances of the methodological approach in considering process as a data modelling primitive.

Three time slices of the simulation are presented in figure 11. The graduated green background represents the digital elevation model, where light green illustrates high elevation (highest elevation is at one bottom right corner) and black low elevation (lowest elevation is at one top left corner). The green nens represent the process of groundwater flow, the blue nens represent Hortonian overland flow, the orange nens represent percolation, and the yellow nens the process of infiltration. No surface ponding occurs in this simulation and at these time steps no saturation excess takes place. From the three time steps in figure 11, the dynamics of the processes can be viewed. Following an initial phase of infiltration, percolation and Hortonian overland flow subsequent to rainfall, the process of groundwater flow dominates in the watershed.

The advantage of a data model that represents process is that it can be queried and analysed. Consequently, insight can be gained as to where and when certain processes dominate, which may lead to a better understanding of the modelled

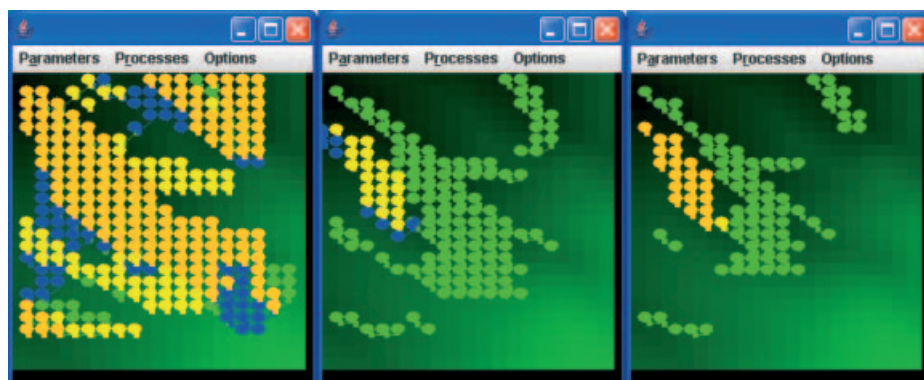


Figure 11. Simulation at three time steps, in progressive order from left to right.

system and give guidance to better ways of interacting with that system. For example, in figure 11 above it is evident that the process of Hortonian overland flow dominates in certain upland parts of Upper Sheep Creek. This is in contrast to typical approaches to modelling that generate results expressing where some energy or mass is at an instant of time within the system, such as water in our watershed, with no information of the processes that caused that state.

With the nen data model the state of processes can be queried, such as asking which process instances or process types have the greatest energy or are moving the most water at a particular time or over a period of time. Because the nen data model represents a process as a spatially extended entity at any moment in time, its dynamics such as changing direction and velocity can also be analysed. Furthermore, the interaction among processes can be explored, such as the use of network analysis to develop new measures of process interaction and extent. Finally, the methodology developed can also provide the testing ground for different definitions of processes, where it is possible to visualize and measure how descriptions of processes within the model compare to known spatial dynamics of processes.

6. Conclusion

The primary methodologies for modelling geographic processes have focused on generating future system or object state representations and analysing these system or object states and the differences between them. An approach presented in this paper furthers our representational capabilities such that process information is explicitly represented and stored with the nen data model. This has the advantage of allowing for exploration into the dynamics of process interactions, explanation of those dynamics, and ultimately of presenting a new epistemological window onto the subject matter. Consequently, as a novel way of simulating the geographic phenomena studied it may provide new insights into how those geographic phenomena operate.

The nen data model provides new avenues for analysis and exploration. Such process analysis not only involves analysis of the results of the simulation with novel analytical techniques suited to the data model, but also analysis of process definitions and how both quantitative and qualitative knowledge might be utilized and tested with the approach developed. It also raises questions of whether other new data models may provide further opportunities for exploring new aspects of well-studied systems and furthering our understanding of those systems.

References

- ABRAHAM, T. and RODDICK, J.F., 1999, Survey of spatio-temporal databases. *Geoinformatica*, **3**, pp. 61–99.
- BENENSON, I. and TORRENS, P.M., 2004, *Geosimulation: Agent-based Modeling of Urban Phenomena* (London: John Wiley and Sons).
- BERNARD, L. and KRUGER, T., 2000, Integration of GIS and spatio-temporal simulation models: interoperable components for different simulation strategies. *Transactions in GIS*, **4**, pp. 197–215.
- BIAN, L., 2000, Object-oriented representation for modelling objects in an aquatic environment. *International Journal of Geographic Information Science*, **14**, pp. 603–623.
- CLARAMUNT, C. and PARENT, C., 2003, Modelling concepts for the representation of evolution constraints. *Computers, Environment and Urban Systems*, **27**, pp. 225–241.
- CLARAMUNT, C., PARENT, C. and THÉRIAULT, M., 1997, Design patterns for spatio-temporal processes. In *Searching for Semantics: Data Mining, Reverse Engineering*, S. Spaccapieta and F. Marganski (Eds), pp. 415–428 (Chapman & Hall).
- CLARKE, K.C., PARKS, B.O. and CRANE, M.P., 2001, *Geographic Information Systems and Environmental Modeling*, Prentice Hall Series in Geographic Information Science (Upper Saddle River: Prentice Hall).
- COVA, T.J. and GOODCHILD, M.F., 2002, Extending geographical representation to include fields of spatial objects. *International Journal of Geographic Information Science*, **16**, pp. 509–532.
- EPSTEIN, J.M. and AXTELL, R., 1996, *Growing Artificial Societies: Social Science from the Bottom Up* (Cambridge: MIT Press).
- ERWIG, M., GÜTING, R.H., SCHNEIDER, M. and VAZIRGIANNIS, M., 1999, Spatio-temporal data types: an approach to modeling and querying moving objects in databases. *Geoinformatica*, **3**, pp. 269–296.
- FRANK, A.U., 2001, Socio-economic units: their life and motion. In *Life and Motion of Socio-Economic Units*, 8 ed. A.U. Frank, J. Raper and J. Cheylan (Eds) (London: Taylor & Francis).
- GOODCHILD, M.F., 2003, The nature and value of geographic information. In *Foundations of Geographic Information Science*, M. Duckham, M.F. Goodchild and M.F. Worboys (Eds), pp. 19–32, (London: Taylor & Francis).
- GRIFFITHS, T., FERNANDES, A., PATON, N.W., MASON, K.T., HUANG, B., WORBOYS, M.F., JOHNSON, C. and STELL, J., 2001, Tripod: A comprehensive system for the management of spatial and aspatial historical objects. In *Proceedings Ninth ACM Symposium on Advances in Geographic Information Systems, ACM-GIS*, pp. 118–123.
- GRUBER, T.R., 1993, A translation approach to portable ontologies. *Knowledge Acquisition*, **5**, pp. 199–220.
- GULYÁS, L., 2002, On the transition to agent-based modeling. *Social Science Computer Review*, **20**, pp. 389–399.
- HÄGERSTRAND, T., 1967, *Innovation Diffusion as a Spatial Process* (Chicago: The University of Chicago Press).
- HUANG, B. and CLARAMUNT, C., 2002, STOQL: An ODMG-based spatio-temporal object model query language. In *Advances in Spatial Data Handling, 10th International Symposium on Spatial Data Handling, 9–12 July 2002, Ottawa, Canada* (Heidelberg: Springer-Verlag), pp. 225–238.
- KAVOURAS, M., 2001, Understanding and modelling spatial change. In *Life and Motion of Socio-economic Units*, A. Frank, J. Raper and J. Cheylan (Eds), pp. 49–62 (London: Taylor & Francis).
- LANGRAN, G., 1993, *Time in Geographic Information Systems* (London: Taylor & Francis).

- LIBOUREL, T., 2001, How do databases perform change? In *Life and Motion of Socio-economic Units*, A. Frank, J. Raper and J. Cheylan (Eds), pp. 155–166 (London: Taylor & Francis).
- MARKS, D., 2001, Introduction to special section: Reynolds Creek experimental watershed. *Water Resources Research*, **37**, p. 2817.
- MERZ, R. and BLÖSCHL, G., 2003, Regional flood risk - what are the driving processes? In G. Blöschl, S. Franks, K. Musiak and D. Rosby erg (Eds), *Water Resources Systems - Hydrological Risk, Management and Development, Proceedings of Symposium HSO2b held during IUGG2003*, July 2003, Sapporo Japan, IAHS Publ. no. 281, pp. 49–58.
- MILLER, H.J., 2004, Activities in space and time. In *Handbook in Transport 5: Transport Geography and Spatial Systems*, P. Stopher, K. Haynes and D. Hensher (Eds) (London: Pergamon/Elsevier Science).
- MOUNTRAKIS, G., AGOURIS, P. and STEFANIDIS, A., 2002, A differential spatio-temporal model: primitives and operators. In *Advances in Spatial Data Handling, 10th International Symposium on Spatial Data Handling*, July 9–12, Ottawa, Canada (Heidelberg: Springer-Verlag), pp. 255–268.
- PANG, M.Y.C. and SHI, W., 2002, Development of a process-based model for dynamic interaction in spatio-temporal GIS. *Geoinformatica*, **6**, pp. 323–344.
- PARKER, D.C. and MERETSKY, V., 2004, Measuring pattern outcomes in an agent-based model of edge-effect externalities using spatial metrics. *Agriculture, Ecosystems, and Environment*, **101**, pp. 233–250.
- PEUQUET, D.J., 1994, It's about time: A conceptual framework for the representation of temporal dynamics in geographic information systems. *Annals of the Association of American Geographers*, **84**, pp. 441–461.
- PEUQUET, D.J., 2001, Marketing space for time: issues in space-time representation. *Geoinformatica*, **5**, pp. 11–32.
- PEUQUET, D.J., 2002, *Representations of Space and Time* (New York: The Guilford Press).
- POUCHARD, L., CINQUINI, L. and STRAND, G., 2003, The earth system grid discovery and semantic web technologies. In *Proceedings of the Workshop of Semantic Web Technologies for Searching and Retrieving Scientific Data, The Second International Semantic Web Conference (ISWC-03)*, 20 October, Sanibel Island, FL.
- RAND, W., BROWN, D.G., PAGE, S.E., RIOLO, R., FERNANDEZ, L. and ZELLNER, R., et al., 2003, Statistical validation of spatial patterns in agent-based models. In *Proceedings of Agent Based Simulation*, 28–30 April 2003, Montpellier, France.
- RAPER, J. and LIVINGSTONE, D., 1995, Development of a geomorphological spatial model using object-oriented design. *International Journal of Geographic Information Science*, **9**, pp. 359–383.
- REITSMA, F. and ALBRECHT, J., 2005, Modeling with the semantic web in the geosciences. *IEEE Intelligent Systems*, **20**, pp. 86–88.
- STEFANAKIS, E., 2003, Modelling the history of semi-structured geographical entities. *International Journal of Geographic Information Science*, **17**, pp. 517–546.
- WACHOWICZ, M., 1999, *Object-Oriented Design for a Temporal GIS* (London: Taylor & Francis).
- WORBOYS, M.F., 2001, Modelling changes and events in dynamic spatial systems with reference to socio-economic units. In *Life and Motion of Socio-Economic Units*, F.A. and R.J. Cheylan (Eds) (London: Taylor & Francis), pp. 155–166.
- YUAN, M., 1996, Temporal GIS and spatio-temporal modeling. In *Proceedings of the 3rd International Conference on Integrating GIS and Environmental Modeling*, Sante Fe, New Mexico, USA, 21–26 January.
- YUAN, M., 2001, Representing complex geographic phenomena in GIS. *Cartography and Geographic Information Science*, **28**, pp. 83–96.
- ZHANG, W. and HUNTER, G.J., 2000, Temporal interpolation of spatially dynamic objects. *Geoinformatica*, **4**, pp. 403–418.