Outline of a Formal Theory of Processes and Events, and Why GIScience Needs One

Antony Galton(✉)

College of Engineering, Mathematics and Physical Sciences, University of Exeter, Exeter, UK
a.p.galton@exeter.ac.uk

Abstract. It has often been noted that traditional GIScience, with its focus on data-modelling functions such as the input, storage, retrieval, organisation, manipulation, and presentation of data, cannot readily accommodate the process-modelling functions such as explanation, prediction, and simulation which it is increasingly acknowledged should form an essential element of the GI scientist's toolkit. Although there are doubtless many different reasons for this seeming incompatibility, this paper singles out for consideration the different views of time presupposed by the two kinds of function: on the one hand, the 'frozen' historical time required by data modelling, and on the other, the 'fluid' experiential time required by process modelling. Whereas the former places an emphasis on events as discrete completed wholes, the latter is concerned with on-going continuous processes as they evolve from moment to moment. In order to reconcile the data-modelling and process-modelling requirements of GIScience, therefore, a formal theory of processes and events is developed, within which their fundamental properties can be made explicit independently of any specific implementation context, and their relationships systematically investigated.

Keywords: Data modelling · Process modelling · Event · Process · Formal theory

1 Introduction

This paper begins (in Sect. 2) with a discussion of the ways in which time enters GIS, through an examination of the various kinds of functions it has been thought desirable for a GIS to perform; here I refer to 'functions' of a GIS in the generic sense of the broad *kinds* of activities that a GIS might enable a user to undertake, as opposed to specific operations such as overlay, interpolation, and generalisation, which are often referred to as GIS functions. I draw a broad distinction between two general classes of functions which, following [26], I call *data-modelling functions* and *process-modelling functions*. These two classes are associated with two distinct ways of viewing time, called 'historical' and 'experiential' after [14], or, perhaps more vividly, 'frozen time' and 'fluid time'.

S.I. Fabrikant et al. (Eds.): COSIT 2015, LNCS 9368, pp. 3–22, 2015.
DOI: 10.1007/978-3-319-23374-1_1

The much-debated question of how data-modelling and process-modelling functions can be integrated into a single system is thus seen to involve as a key component the integration of the two corresponding approaches to time. It appears that while the historical view of time is dominated by events, the experiential view is dominated by processes, and hence a successful integration of the two depends on a correct understanding of processes and events, and how they are related. The main purpose of this paper is, having established the necessity for this in the context of GIS, to undertake the initial development and formalisation of a robust and highly general theory of processes and events. Sect. 3 is devoted to a close analysis of the notions of 'process' and 'event' and the concepts needed to handle their interrelationships. The discusssion is informal, but is backed up by a formal development, outlined in the Appendix. No claims are made for completeness of the theory: it is unashamedly a first step, which will require further detailed elaboration before it can fully serve its purpose as a standard reference benchmark for the proper treatment of time in GIS.

2 GIS Functions, and Two Approaches to Time

What are the functions of a GIS? It is usual to draw a contrast between 'traditional GIS functions' comprising the input, storage, retrieval, organisation, manipulation, analysis and presentation of data, and a range of more advanced capabilities such as explanation, simulation and prediction which engage with the data through some form of theoretical understanding of the real-world systems and processes that the data represent.[1] This contrast has been described in various different ways, all tending to the same (sometimes rather despairing) conclusion that it is high time the data-modelling functions of GIS were integrated with the process-modelling requirements of at least a substantial proportion of GIS users. A representative sample of sources in which such statements can be found is [5–7,12,20,24,26,29,31].

These two sets of functions seem to point to two rather different kinds of system. On the one hand, there are those systems, lying somewhere on a continuum between a digital map and a spatial database, which encompass the descriptive and representational functions of a GIS, and on the other, there are systems which encompass the exploratory functions such as prediction and simulation that are increasingly thought of as natural adjuncts to a GIS. This distinction has appeared in the literature under a variety of different names, including 'map-representation systems' vs 'reality-representation systems' [20], 'data models' vs 'process models' [26], 'history models' vs 'process models' [24], and 'information systems' vs 'modelling systems' [12].

[1] The term 'analysis' could perhaps be included with the second set of functions as well: it is a broad term which covers a range of different activities. However, many traditional GIS functions such as interpolation, overlay, and generalisation are often described as 'analytical', and many, though not all, of the functions described by O'Sullivan and Unwin in their book on Geographic Information Analysis [23] belong with the 'traditional GIS functions' rather than the 'more advanced capabilities'.

Traditionally, these two kinds of functions — data-modelling functions and process-modelling functions — have been handled separately, the former by a general-purpose GIS and the latter by a special-purpose system designed to meet the specific requirements of a particular application area such as meteorology, geomorphology, animal ecology, traffic systems, or human population studies. The significant question is how to get the two systems to "talk to each other", so that the results of the calculations performed by the process-modelling system can be made available to be stored, manipulated, and output in human-friendly form by the GIS. Many of the authors cited above have asked this question and explored the ramifications of different ways of answering it; but already 20 years ago, Raper and Livingstone [26] suggested that the question was outdated, and that 'the next step should be the fusion of models and spatial representations within new object-oriented environments and *not* the integration of incompatible systems which force representational compromises'.

Of particular relevance to all this is the role of *time* in geographical representation. Time may, but need not, be involved in spatial data modelling, but it is almost invariably of central importance in spatial process modelling. It is significant, however, that while both kinds of modelling may need to work with time, they work with *different kinds of time*. What do I mean by this?

Considering the data-modelling functions first, it is generally accepted that the basic unit of geographical information is a combination of place (*where?*), time (*when?*), and theme (*what?*): In place p at time t there is X. This is Peuquet's *Triad Framework* [24].[2] The basic schema covers a multitude of variations — for example, p can be a point, a grid square, or a region; t can be an instant, a 'standard' interval such as a calendar month or year, or an 'arbitrary' interval; and X can be a value (of a field), an object (which could be a fixed physical feature, something mobile, a social or political unit, a collective, ...), or a process or event — but in essence a GIS consists of a repository of such triples together with a set of algorithms for manipulating them in accordance with the data-modelling functions listed above.

Time is often said to enter this picture in two different ways [28]. There is the time in which the manipulations referred to above occur, known in the temporal database community as *transaction time*; insofar as it is recorded in the database itself it belongs to the metadata associated with the geographical data, indicating, for instance, when a particular triple was entered into the repository, or when it was superseded. This time is concerned with the history of the database itself, and has nothing to do with the temporal dimension of the geographical reality being described. Much more significant, for our present purposes, is the so-called *valid time*, which is the time referred to by the triples, the time in the real world at which the state of affairs described by the triple actually obtained. This time records the history of the geographical reality that the system is being

[2] In some more recent treatments, place and time are amalgamated, and the nature of the theme is made more explicit, as in the *geo-atom* of Goodchild *et al.*, which takes the form $\langle \mathbf{x}, Z, z(\mathbf{x}) \rangle$, where '$\mathbf{x}$ defines a point in space-time, Z identifies a property, and $z(\mathbf{x})$ defines the particular value of the property at that point' [18].

used to record, and the presence of a triple (p, t, X) in the repository amounts to an assertion that there really was X in place p at time t — if this was not the case, then the repository is in error and stands in need of correction.

Turning now to the process-modelling functions, we find a rather different picture. Rather than being a repository of facts and data-processing algorithms, a process-modelling system might be thought of as a repository of *theories*, that is, theoretical models of the laws and regularities that are believed to hold sway in the world. These can be used to *generate* a picture of the world — which usually means a picture of the world as it evolves through time. This is 'spatial process modelling', where I am using this term in a generic sense to cover a multitude of different possible formalisms such as numerical solution of partial differential equations, cellular automata [4,9,30], GeoAlgebra [29], agent-based models [7], geographic automata [32], and doubtless many others as yet unthought of.

Spatial process modelling can be used in several different ways, notably:

- *Prediction*: Starting from known present data X, run the model to predict future data Y.
- *Explanation*: Starting from known past data X, run the model to 'predict' present data Y, comparing the result with known present data Z. If Y and Z agree, the model is accepted as providing an explanation for Z, if not, it is rejected and a new model tried.
- *Retrodiction*: Starting from hypothetical past data X, run the model to predict present data Y — if Y agrees with known present data Z, the hypothesis X is corroborated.
- *Planning*: Starting from hypothetical near-future data X, run the model to predict future data Y. If Y agrees with some desired future goal Z, use X as a plan in order to achieve Z.

It is noteworthy that, throughout these examples, (p, t, X) triples are not being asserted but hypothesised, put up for consideration as possibilities — the grammatical mood here is in effect *subjunctive* rather than, as in the data-modelling system, indicative; and this is related to a key distinction between the ways in which data modelling and process modelling relate to time.

The (valid) time of a data-modelling system is *passive*, exactly comparable to space. Here time is "just another" form of space, another coordinate in the multidimensional presentation of data. Time in such a system is as it were static. Insofar as processes and events are represented in the system, they are portrayed as "frozen" in time, inactive, merely more bits of data. In contrast, the time of a process-modelling system is *active*. When running a simulation, events and processes are *enacted* in symbolic form within the system — this is obvious in the case of a real-time simulation which we see unfolding before our eyes, but even in the case where calculations are performed to derive an end result from data pertaining to some earlier time, simulation is taking place in a more abstract sense, and we can say that here too processes are being enacted. This is "fluid" time in the sense of something that flows, in the way that we customarily (albeit metaphorically) conceive of time as doing, and changes really occur.

The results produced by running a process model in such a system are by nature hypothetical. They are not data in the sense of something *given* from outside (for example, entered by the user in the belief that they truly represent the world out there); rather, they are derived within the system. That is why they can be usefully *compared* with real data, as in the scenarios envisaged above. But time is also fluid for a real-time monitoring system, such as a collection of sensors gathering data about real-world processes to send to a data-modelling system for analysis. From the point of view of the latter, it is immaterial whether the data it is given to work with are factual or hypothetical: the same data-processing techniques can be applied to both — after all, hypothetical data are of little use if they do not resemble factual data to the extent that they can be considered as *possibly* factual.

In process modelling and process monitoring, time is modelled *as* time, that is, the temporal sequence in which the represented events are handled computationally corresponds to the sequence in which they occur in the hypothetical world that is being modelled or the actual world that is being monitored. In the case of process modelling, one might say that this is because that world is *in* the machine, so its time *is* the machine's time. In data modelling, by contrast, the time of the world being modelled is captured, not through temporal sequence in the data processing, but symbolically as values on a coordinate axis, which may be stored and processed in any order, just like the values on the spatial coordinate axes. In process modelling, the processor itself might be said to *experience* the time it is modelling, whereas in data modelling it merely *records* it. In [14], I drew a contrast between the 'experiential' and 'historical' accounts of the world, and related these to the distinction between processes and events, processes being concerned with the low-level goings-on that are the immediate objects of experience, events to more synoptic summaries of salient aggregates of processes as they are recorded in the memory. This distinction closely matches that drawn here between "fluid" time and "frozen" time respectively.

The relationship between processes and events is crucial here: the process-modelling system (or, in a different way, a real-time monitoring system) generates ongoing processes, and information about these processes has to be passed in some form to the data-modelling system. But it is, presumably, the responsibility of the latter to extract from this processual flux those hard nuggets of salience which constitute events, and which from a human perspective represent information rather than mere data.

Simulation systems and real-time monitoring systems are points of contact with fluid time: either the time that actually elapses in the real world (and of course it elapses by virtue of the processes going on in the world — there is no need, here, to invoke a Newtonian notion of absolute time flowing independently of anything that happens), or simulated time, which elapses by virtue of the computations driving the simulation. Our *experience* of the world is like this, too: a direct engagement with fluid temporal processes — hence 'experiential' as the designation for the latter. On the other hand, our representation and reasoning about the world displays a strongly event-oriented bias: events, that is salient discrete "chunks" of happening, which can be labelled as individuals and

marshalled into networks of cause and effect, form the basic subject-matter of our temporal discourse, and as such, they, rather than processes, have been the focus of many important proposals for how time should be handled in GIS, here conceived primarily as a data-modelling system [21,25,34]. What we most want from a temporal information system is information about *what happens*, and that primarily means information about events. It follows that if a process-modelling system is to be successfully integrated with an event-oriented data-modelling system, then we need a robust account of how processes are related to events.

In the remainder of this paper, I shall start to develop a basic theory of processes and events from an informatic perspective, laying down the fundamentals of a logic of occurrence which underlies the presentation of temporal facts. Whatever other specialised apparatus is employed by a GIS for recording and manipulating such facts, I would maintain that it should be founded on a secure logical bedrock of this kind. Without such a foundation, talk about processes and events will continue to be subject to confusion as different researchers use the terms in their own way without any kind of agreed common standards. One thing that emerges from the work presented below is that even with regard to the simple logic of occurrence, there are already some formidable difficulties which must be overcome by carefully distinguishing different varieties of process and event and how they are related. Either that, or sweep the difficulties under the carpet and risk tripping over them later to fall flat on one's face.

3 Towards a Formal Theory of Processes and Events

A trip is an event, whereas travel is a process. [1]

There is little agreement on how to use the terms 'process' and 'event': Worboys [33] speaks of an 'astonishing variety of usage and definition', and notes that 'One person's process is another's event, and vice versa'. But whereas Worboys, in that paper, declines to pursue the matter further, I believe that we can and should strive to achieve a common understanding of the issues at stake here, which are not just a matter of terminology but strike deeply into the conceptual foundations of how we represent and reason about the world.

One of the problems is that there seem to be two fundamentally different meanings of the word 'process', and I believe that a good deal of the confusion surrounding the use of this term is due to the conflation of these two meanings.

On the one hand, the word 'process' is used to denote an activity that is not intrinsically bounded and which can, at a sufficiently coarse temporal scale, be conceptualised as homogeneous. Processes in this sense include the flowing of a river, cliff erosion along a stretch of coastline, the year-on-year growth of a tree, the gradual encroachment of built-up area into the countryside surrounding a city, the movement of traffic along a street, continental drift, as well as human activities such as walking, talking, eating, swimming, and travelling. This kind of process contrasts strongly with the notion of 'event', which prototypically refers to an intrinsically bounded, discrete occurrence which may, at a sufficiently

coarse temporal scale, be conceptualised as point-like.[3] Examples of events in this sense include the collapse of a particular chunk of cliff, the falling of a tree (whether through human or natural agency), the construction (or destruction) of a house, a volcanic eruption or an earthquake, a journey from A to B, and human actions such as a walk from home to the office, utterance of a sentence, eating an egg, or swimming a length.

The second use of the word 'process' is to denote a structured closed routine leading to a specified end point. Processes of this kind typically involve human (or animal) agency, and are often described as 'the process of *X*ing a *Y*'. Examples include the processes of making a pot of tea, registering for a conference, booking a train ticket, constructing a window-frame, building a [bird's] nest, and spinning a [spider's] web. Processes of this kind are typically governed by a specific *procedure* which may be specified in advance.[4] Each enactment of the procedure is in fact an event (that is, it is intrinsically bounded and discrete).[5] Moreover, this event is composite: that is, it is built up out of sub-events corresponding to the various phases of the procedure. As such, this kind of process cannot readily be described as homogeneous.

From the above, it is evident that the two kinds of process are radically different from one another. Processes of the first kind are homogeneous (unstructured) and not intrinsically bounded, whereas those of the second kind are structured and intrinsically bounded. On the other hand, it can hardly be regarded as merely coincidental that the same word is used to describe both of them. What is the connection? I shall defer discussion of this until later, but meanwhile, in order to emphasise that there are two very different kinds of phenomenon here, I shall reserve the term 'process' for the first kind — that is, the open-ended ongoing process conceptualised as homogeneous — and call the second kind 'routines'.[6]

Here I want to pick up my earlier use of the phrase 'at a sufficiently coarse temporal scale'. Scale, or *granularity*, is all important here in enabling us to arrive at a worthwhile conception of the relationship between processes and events. One reason for this is that scale is closely related to *aspect*, which is concerned with the different points of view from which one and the same thing can be considered: specifically, in the case of something going on in time (an occurrent), whether we are concerned with the occurrence as a whole, including a beginning and end — and in this case, whether we are primarily interested in the beginning or the end, or perhaps the state resulting once the end has occurred — or, alternatively, with what is going on from moment to moment, how the occurrence presents itself at the point of experience or recording.

[3] Cf. [10]: 'An event is an individual episode with a definite beginning and end . . . '.

[4] These are similar to what Aitken and Curtis [3] call Scripts: 'A Script is a typical pattern of events that can be expected to re-occur: "dining in a restaurant" and "brushing one's teeth" being well known examples' (the restaurant example comes from the original exposition of the Script concept by Shank and Abelson [27]).

[5] Cf. [33]: '[C]omputational processes are rather like computer programs, which when executed result in occurrents'. Here it is the program execution itself that is described as an occurrent, not the outputs resulting from it.

[6] In [14], these are called 'open' and 'closed' processes respectively.

To illustrate this with a simple example, consider a succession of bursts of machine-gun fire. Taken together as whole, we might describe this as an event, which begins at the start of the first burst and goes on till the end of the last one. But putting ourselves in the position of someone experiencing this, either as the gunner, his intended victims, or an onlooker, it seems natural to describe it as a process: an ongoing process consisting of one burst after another, with perhaps no indication of when or whether it is going to end. Each individual burst, on the other hand, is clearly an event, with a clear-cut beginning and ending, and the larger-scale process consists of an indefinite number of repetitions of events of this kind.[7] Now turn up the temporal "magnification" to examine the structure of each burst. What it is, is simply a "chunk" of machine-gun fire; here, machine-gun fire is a process. In principle it can go on indefinitely in the same way (hence, unbounded and homogeneous), though in practice any instance of machine-gun fire will have a beginning and an ending, and if we include these in our description of it then what we have is, precisely, a "chunk" of that process; and this is an event. Ignore the beginning and end now, and concentrate on the process of machine-gun fire: on closer examination (i.e., stepping up the temporal magnification again), we see that it consists of a sequence of events, each of which is the firing of an individual cartridge. Each of these firing events can itself be examined more closely to reveal various lower-level processes and events which go to make it up.

This way of looking at the relationship between processes and events can be used to reconcile two rather different views of events that have appeared in the literature. On the one hand, Yuan [35] regards an event as 'a spatial and temporal aggregate of its associated processes', and states that 'a process is measured by its footprints in space and time'. In relation to precipitation, the subject of her case study, she notes that 'an event marks the occurrence of precipitation' whereas 'a process describes how it rains'. Although her understanding of 'process' and 'event' are somewhat different from what is proposed here, the notion that events can be built up (or 'assembled', to use Yuan's word) from processes represents a point of commonality. Contrasted with this is the notion of events as marking points of discontinuity in an otherwise smooth course of history, as expressed, for example, by Langran and Chrisman [21], who portray events as effectively instantaneous transitions between preceding and succeeding states of affairs. Here there is no indication that events may themselves comprise extended episodes within which various processes occur. Our example above, however, shows that these two seemingly very different views of events are quite compatible, and merely reflect different granularities at which events can be portrayed.[8]

[7] Note: This must be construed carefully: it is the *type* of event that is repeated, each individual event occurs just once.

[8] It is instructive in this connection to compare Fig. 2 in [21] with Fig. 1 in [35], focussing particularly on the role assigned to the term 'Event' in the two diagrams.

This example has highlighted two general *temporal operators* which can be used to define event-types in terms of processes or vice versa.[9] They are:

- *Chunking*: For a process P, we define an event-type $chunk(P)$, each of whose occurrences consists of P starting, going on for a while, and then stopping. In an information system, an event of type $chunk(P)$ might be constructed by selecting a time-interval $[t_1, t_2]$, and "filling" it with a "texture" corresponding to process P, analogous to selecting a spatial region and filling it with some value such as land-cover type. Note that the *boundary* is not filled: P is not active at the endpoints of an interval on which a chunk of P occurs.
- *Repetition*: For an event-type E, we define a process $rep(E)$, which consists of an indefinite number of occurrences of E in (sufficiently quick) succession. (Here 'sufficiently quick' will depend on the specific event-type involved, and the context in which we are considering it — more on this below).

It should be emphasised here that chunks are to be understood as maximal: as we are using the term, we cannot pick out a day's worth of the Earth's rotation and call this a chunk of rotation.

In the literature, the term 'process' has been used to refer both to processes as we understand them here, and to *chunks* of process. For example, when Yuan [35] describes an event as an 'aggregate of its associated processes', she must mean process chunks, but when she speaks of a process as 'a continuing course of development', the word 'continuing' seems to rule out the chunk interpretation.

What other temporal operations are there? Up to now I may have given the impression that any event-type must be specified as a chunk of some process, but this is not correct. Some events are directly composed of other events, where these constituent events are not sufficiently homogeneous to be regarded as forming a process of type $rep(E)$ for any E. As an example, consider an event in which someone refuels their car: this event consists of a sequence of subevents, namely: drive into the petrol station; if necessary wait in the queue; draw up alongside the petrol pump; switch off the engine, get out of the car, unscrew the cap to the fuel tank; etc., etc. This is, of course, an enactment of a routine, in the sense introduced above. There is no single process of which this event is a chunk. What is needed here is a direct event-composition operator, which combines a sequence of events into a single larger event. It is standard to denote such an operator ';', so the event-type defined as the *sequential composition* of event-types E_1 and E_2 is denoted $E_1; E_2$. If this operator is stipulated to be associative, then any expression of the form $E_1; E_2; \ldots; E_n$ is unambiguous; but as will be shown below, such a stipulation may be problematic, in which case we must distinguish differently-bracketed variants such as $E_1; (E_2; E_3)$ and $(E_1; E_2); E_3$.

[9] It is important to note that the general theory has to handle event-types rather than specific unique occurrences. In defining what is meant by a chunk of some process, for example, we are characterising a type of event, not an individual event. There may be many different individual occurrences which come under this description (or only one, or none), whereas an individual event is by nature unique. If we say 'It happened twice' or 'It happened again', by 'it' we can only mean an event-type, of which we are reporting another occurrence.

With events it is natural to consider sequential composition, because events have both beginnings and endings, and therefore one can readily locate the beginning of one event at or just after the end of another, the two together thereby forming a candidate for being considered as constituting a larger event. A process, on the other hand, does not intrinsically have a beginning and an ending; as soon as a process is considered together with its endpoints, it is being treated as an event (that is, a chunk of process). For this reason it does not seem possible to define sequential composition of processes as such. On the other hand, it is natural to consider the *parallel* composition of processes: that is, two processes whose simultaneous operation is regarded as constituting a process in its own right. An example, on a small scale, would be someone driving a car while speaking on their mobile phone — in this case two individually permitted activities become illegal in (parallel) combination — and on a larger scale, the climate becoming both warmer and wetter.

Parallel composition of events is also possible, although it is conceptually more complex, since one has to specify whether the events should begin together, end together, or both — or merely overlap without any coincidence of endpoints. A range of different possible operators might be suggested here; experience with different application contexts might single out some as especially useful.

Various forms of sequential and parallel composition are widely encountered in the literature, forming essential components of algebras or calculi that have been proposed for different purposes. Examples include the event-composition operators used in Active Databases [2,17], Dynamic Logic [19], and Artificial Intelligence [13]. Most such systems only handle events (while sometimes using the term 'process' to denote them, the focus being on routines rather than processes). As a result, the idea of having operators mapping *between* states and processes is more rarely encountered. An exception is in linguistics, where attempts have been made to formalise the semantic relationships amongst different verbs or verb-phrases, and the expression of such relationships through the linguistic phenomenon of aspect (notably perfective *vs* imperfective), as for example 'it is raining' refers to a process but 'it rained three times yesterday' refers to three occurrences of the event-type which we here describe as a chunk of raining. Operators mapping between events and processes or states and vice versa are discussed by, amongst others, [11,22].

Comparable constructs can be found in GIScience, although it is rare to find explicit formalisations of them. Yuan [35] has a section on 'Assembling Events and Processes' which includes a rule that is analogous to chunking (to determine when 'the rain areas in T_1 and T_2 belong to the same rainstorm process'), and a composition rule that builds a precipitation event from overlapping or sequential chunks of precipitation processes which form an unbroken extent. Claramunt *et al.* [8] similarly consider aggregations of process chunks (here just called processes) to form larger 'STP composites'. Worboys and Hornsby [34] discuss the combination of events into 'temporal sequence aggregations', e.g., the sequence of `PlaneLanding` followed by `PlaneTaxiToGate`, followed by `Passenger Deplaning`, but again, no attempt is made to formalise the properties of such sequences. Although the necessity for such operations is frequently

acknowledged, and attempts have been made to systematise their definition and behaviour, rigorous formalisations such as we present here seem to be lacking.

The example 'It is raining' cited above leads us to consider a further possible operator, since if we regard the English continuous tense (i.e., the form 'be ... ing') as expressing a process of which the simple form ('It rained') reports an occurrence of a chunk, then it would seem we should be able to apply this to *any* event type to yield a process of which each instance of that event type is a chunk. If we apply this to a routine such as making a pot of tea, we arrive at the idea that each enactment of this routine (that is someone making a pot of tea on one occasion) can be considered to be a chunk of a process called 'making a pot of tea'. In reality, of course, the various stages of making a pot of tea are very different from each other (boiling the water, putting the tea in the pot, pouring the boiling water into the pot, etc.) so we cannot really say there is a single homogeneous activity such that making a pot of tea just involves engaging in that activity for a certain period of time. We can, however, regard the continuous tense as supplying a blanket term to cover all the activities involved at each stage of making a pot of tea, conceptualising them as forming a single notionally homogeneous activity, unified by the fact of their forming part of the larger event: the complete event is then indeed a chunk of that activity. It may be that something like this is at the root of the use of the term 'process' to refer to a closed routine (our second sense above): it refers to the activity involved in executing such a routine, glossing over the fact that this activity may be a complex heterogeneous compound of individually homogeneous subprocesses. This operation, by which a 'higher-level' process is created from an event, can be called *dechunking*.

The operators suggested so far look as though they should form the basis of a formal calculus of processes and events, but the matter is not entirely straightforward. Here I shall discuss some of the issues in an informal way; in the Appendix can be found partial formalisations which highlight where the problems are.

It seems natural to suppose that the *chunk* and *dechunk* operators should be mutually inverse, so that

- for any event type E, $chunk(dechunk(E)) = E$, and
- for any process P, $dechunk(chunk(P)) = P$.

A natural way of defining these two operators is as follows:

- There is an occurrence of event type $chunk(P)$ on interval $[t_1, t_2]$ so long as P is active throughout (t_1, t_2) but not at the endpoints t_1 and t_2.
- The process $dechunk(E)$ is active at time t so long as there is an occurrence of E on some interval $[t_1, t_2]$ such that $t_1 < t < t_2$.

Note the style of these definitions: an event type is defined by providing its *occurrence conditions*, that is, necessary and sufficient conditions that an event of the specified type occurs over a given interval, whereas a process is defined by providing its *activity conditions*, that is, necessary and sufficient conditions that the specified process is active at a given time.

With these definitions we can only obtain the result that *chunk* and *dechunk* are mutual inverses (Appendix, Theorems 3 and 4) if we postulate that:

1. Distinct occurrences of a given event type cannot overlap. In this case we shall call the event type *discrete*.
2. Processes are active on open intervals, i.e., if P is active at time t then there are times $t_1 < t < t_2$ such that P is active throughout (t_1, t_2).
3. A process cannot persist indefinitely into the future or have persisted indefinitely far into the past; that is, at any time there are both earlier and later times at which the process is not active. In this case we shall call the process *locally finite*.[10]

We also have to make a number of assumptions about the temporal ordering; for most purposes it suffices that the ordering is irreflexive, transitive, linear, and dense — but in one place we also need to assume that it is continuous.

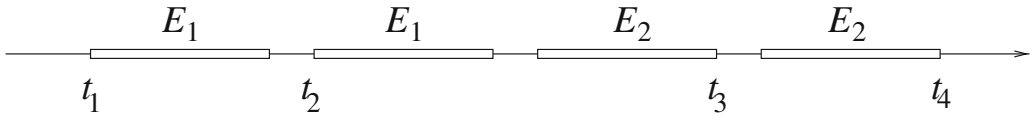
Some of these principles are questionable, and we can easily find geographical examples which might lead us to question them. The discreteness principle seems particularly vulnerable. So long as the event type is sufficiently narrowly defined, we can enforce discreteness, but very often if we try to broaden the definition to create a more general event type, overlapping becomes possible. Consider the event type 'flight by A', where A is a particular individual aircraft. Clearly, distinct occurrences of this event type cannot overlap, since an aircraft cannot be engaged in two flights simultaneously. But if we broaden the event type to 'flight by any aircraft', then over the world there are thousands of individual occurrences of this event type in progress at any moment. Similarly, if the event type is 'rainstorm here' (where 'here' denotes any sufficiently small region), then overlapping occurrences are ruled out since it is not possible for two rainstorms to be in progress at the same place at the same time, whereas the broader event type 'rainstorm anywhere' is again one with many temporally overlapping occurrences. If for technical reasons we wish to maintain the discreteness principle, for example in order to preserve the mutually inverse character of the chunking and dechunking operations relating events to the processes they comprise, then we will have to outlaw general event types such as 'flight by an aircraft' or 'rainstorm anywhere'.

Likewise one may wish to question the local finiteness principle for processes. As an example, consider the rotation of the Earth. So long as it has existed, the Earth has rotated, and no doubt it will continue to rotate for as long as it exists. It is of course true that there are times in the distant past when the Earth did not exist, and at these times the Earth's rotation was not active; and in the (we hope distant) future there will be times when the Earth no longer exists. Thus strictly speaking this process is locally (indeed globally) finite; but from the point of view of a GIS, one would naturally wish to ignore this very long-term perspective and treat the rotation of the earth as a constant backdrop to the events and processes one wishes to describe.

[10] As distinct from 'globally finite', which would mean there is a time before which the process is never active, and a time after which it is never active.

Turning now to the sequence and repetition operators, we find that they too present problems when we try to formalise them. Consider for example how we might formally specify the conditions of occurrence for an event type of the form $E_1; E_2$ — to be concrete, say, an earthquake followed by a landslide. An occurrence of this event type must consist of an occurrence of type E_1 followed by an occurrence of type E_2. What exactly is meant by 'followed' here? How soon after the earthquake must the landslide occur in order for the two together to count as 'an earthquake followed by a landslide'? The strictest requirement would be that the earthquake must be followed immediately by the landslide; that is, for times t_1, t_2, t_3, the earthquake occurs on the interval $[t_1, t_2]$ and the landslide occurs on the interval $[t_2, t_3]$. But we do not normally insist on this, and indeed it often does not make sense to do so. One of the reasons we may be interested in events of type $E_1; E_2$ is because we are interested in the possibility of a causal connection between the components. Such causal effects may well operate with a delay, which may be rather long in some cases. It is not usually possible to specify a precise upper limit to the length of time that must elapse between an occurrence of E_1 and an occurrence of E_2, though one might specify an imprecise limit and build this into the definition of the compound event type.

There is another difficulty, however. To illustrate this, consider the case:



Here two separate occurrences of E_1 are closely followed by two occurrences of E_2. How many occurrences of $E_1; E_2$ are there? A liberal interpretation might define the occurrence condition for $E_1; E_2$ as follows:

> There is an occurrence of $E_1; E_2$ on interval $[t_1, t_2]$ if and only if there are times $t_1 < t \leq t' < t_2$ such that there is an occurrence of E_1 on $[t_1, t]$ and there is an occurrence of E_2 on $[t', t_2]$.

On this liberal interpretation there are four occurrences of $E_1; E_2$ in the case illustrated above, occupying the intervals $[t_1, t_3]$, $[t_2, t_3]$, $[t_1, t_4]$, and $[t_2, t_4]$, which means that $E_1; E_2$ is not discrete.

If we want to avoid this consequence, we must restrict the occurrence condition for $E_1; E_2$ in some way. If our goal is to secure discreteness then it seems that the least we can do is the following:

> There is an occurrence of $E_1; E_2$ on interval $[t_1, t_2]$ if and only if there are times $t_1 < t \leq t' < t_2$ such that E_1 has an occurrence on $[t_1, t]$ but no other occurrence starting within $[t_1, t_2]$, and E_2 has an occurrence on $[t', t_2]$, but no other occurrence ending within $[t_1, t_2]$.

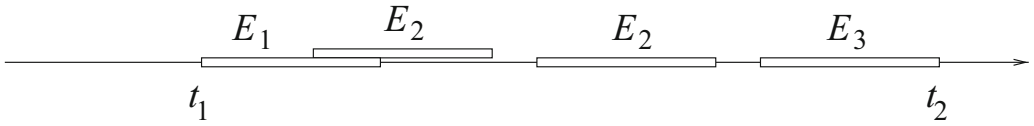
Under this restricted interpretation, the only occurrence of $E_1; E_2$ in the scenario illustrated above is the one on $[t_2, t_3]$. It can be proved (Appendix, Theorem 6) that thus defined, an event type of form $E_1; E_2$ is discrete so long as E_1 and

E_2 are. This is a desirable consequence if we wish the chunking and dechunking operations to be mutually inverse for all event and process types.

There is a disadvantage to this manoeuvre, however. With our first, simple, definition of $E_1; E_2$ we can prove (Appendix, Theorem 5) that sequential composition is an associative operator, that is, that

$$E_1; (E_2; E_3) = (E_1; E_2); E_3.$$

The advantage of this is that it gives us for free, as it were, a definition of threefold sequential composition, which we would naturally write as $E_1; E_2; E_3$. The restricted definition of $E_1; E_2$, on the other hand, is not associative. This is easily seen from an example such as the following:



Here, according to the restricted version of sequential composition, there is an occurrence of $E_1; (E_2; E_3)$ on $[t_1, t_2]$, but not of $(E_1; E_2); E_3$. In such a case it is less clear what we might mean by $E_1; E_2; E_3$.

Thus we have a broad interpretation of ';' which is associative but not discrete, and a narrow interpretation which is discrete but not associative. It is natural to ask whether some compromise interpretation can secure both associativity and discreteness, but it would appear that as we move from our broad interpretation in the direction of the narrow interpretation by adding gradually more stringent conditions, we lose associativity before we gain discreteness.[11]

It should be noted, however, that overlapping is not always undesirable — we have already seen that requiring discreteness can restrict the level of generality of allowable event-type descriptions. Another case where overlapping should be allowed concerns repetition. What if we form the sequential composition of an event type with itself? An occurrence of $E; E$ consists of two occurrences of E, one after the other. If now we have *three* consecutive occurrences of E, then the first two constitute one occurrence of $E; E$ and the second and third constitute a second, overlapping with the first. Of course the three occurrences of E together also constitute an occurrence of $E; E; E$. If we allow such composite events, then we can easily define the repetition process, $rep(E)$ as $dechunk(E; E)$ — or if we require a minimum of, say, five occurrences of E to count as a process of repetition, then $dechunk(E; E; E; E; E)$.

In view of these considerations, we should not insist that event types are discrete, say, or that event composition must be associative (and similarly with other properties one may wish to enforce in some cases), but rather use discreteness to define a specific subclass of events, which one may invoke as the occasion demands. Then for example we can state that if E is discrete then so is

[11] I have not proved this; it is a conjecture based on experiments with a number of plausible candidate definitions.

$chunk(dechunk(E))$, and that if P is locally finite then $dechunk(chunk(P)) = P$. In other words, the main theorems of the formal theory will be conditional in form. This leaves it open to the user of the theory to decide whether they want all events to be discrete, say, or only events of a certain type. In general, if an event type is precisely defined by means of its occurrence condition, then this is already sufficient to determine whether it is discrete.

A more general moral is this. It seems obviously desirable, for the purposes of building a robust and reliable computational infrastructure for a temporal GIS, to give precise generic specifications of the properties and relationships pertaining to the most basic patterns of temporal phenomena. However, as soon as we try to do this in a rigorous and comprehensive way, we find the enterprise to be fraught with difficulties, and it becomes far from clear exactly how these basic notions should be defined, or what the consequences of defining them in any particular way may be. For this reason, unpalatable as it may be from the perspective of GIS developers eager to proceed quickly to a stage where they can begin working with concrete applications, a good deal of logical or mathematical "spade work" has to be done before we can deliver a product with the required degree of reliability. Here we have been able to describe (and, as outlined in the Appendix, execute) only a small portion of such spade work.

Beyond this, a further requirement is to develop, within the formal framework outlined here, a way of representing causal relations amongst states, processes and events. These causal relations would not in themselves constitute an explanatory theory, but would provide an interface through which the relationships determined by means of causal rules within the process-modelling system can be fed back into the data-modelling system. The basic qualitative causal vocabulary consists of, in addition to 'cause' itself, such terms as 'perpetuate', 'maintain', 'allow', 'enable', 'prevent' and 'disable'. In everyday speech these terms may be used in a variety of ways; for the purpose of a more disciplined treatment we need to determine exactly what relations we want to refer to, and to select appropriate vocabulary to describe them. For more on this see [15,16].

4 Conclusion

Following a long-standing tradition in GIScience, this paper began by focussing on the distinction between two fundamentally different classes of function that it has been thought desirable for a GIS to accommodate: on the one hand the data-modelling functions that are concerned with tasks such as the input, storage, retrieval, organisation, manipulation, and presentation of data, and on the other hand the process-modelling functions such as explanation, simulation and prediction which, to be successful, must embody a theory of the real-world phenomena which are being modelled.

It was noted that these two kinds of function, and the systems in which they are implemented, treat time in different ways: for the purposes of data modelling, time is regarded as another static dimension like those of space, providing another coordinate dimension for indexing thematic elements. This is what I

have called 'frozen' or 'historical' time. By contrast, for the purposes of process modelling, it is necessary to take seriously the dynamic nature of time, enabling data to be collected on the fly or generated hypothetically by means of various kinds of process simulation. This leads to a view of time which I have labelled 'fluid' or 'experiential' time. While historical time places an emphasis on completed events, experiential time is more concerned with ongoing processes.

As has previously been acknowledged, the integration of data-modelling functions and process-modelling functions gives rise to considerable difficulties. There may be many reasons for this, but the one I have focussed on in this paper is precisely the discrepancy between the divergent approaches to time that are required by the two kinds of function, exacerbated by the lack of a principled theory of temporal phenomena from an informatic perspective. In the second part of the paper, the foundations for such a theory were laid down: a clear distinction between events specified by occurrence conditions and processes specified by activity conditions, as well as operations for deriving events from processes (chunking), processes from events (dechunking, repetition), and events from events (sequential composition), thus providing a formal framework within which processes and events can be accommodated within an information system.

The exposition of the formal theory has not addressed the issue of how it might be implemented in a working system. While this may be perceived by some as a weakness, I believe that, on the contrary, it is essential. We are talking about the fundamental structure of some of our most basic temporal concepts; any specific implementation must inevitably include many details, concerning for example the data structures used for representing different elements of the theory, which might be specified in many different ways compatibly with the underlying theory and which, being essentially irrelevant to that theory, would only serve as a distraction and, perhaps, be accorded undue importance. Having been developed in a clean, implementation-independent way, the theory can then stand as a benchmark, or reference standard, against which many different implementations in specific systems may be assessed.

A Notes Towards a Formal Theory of Processes and Events

All the theorems listed below have been proved, but there is no space here to include the proofs. These may be obtained from the author on request.

We define a many-sorted first-order language with identity, with sorts $\mathcal{P}$ (Processes), $\mathcal{E}$ (Event types), and $\mathcal{T}$ (Time instants). We could have introduced an additional sort for time intervals, but instead we will refer to an interval by means of a pair of instants, representing its beginning and end points.

The primitive predicates are:

- *Active*, of type $\mathcal{P} \times \mathcal{T}$, where $Active(P, t)$ means that P is on-going at t.
- *Occurs*, of type $\mathcal{E} \times \mathcal{T} \times \mathcal{T}$, where $Occurs(E, t_1, t_2)$ means that an event of type E occurs on the interval $[t_1, t_2]$.

– $<$, of type $\mathcal{T} \times \mathcal{T}$, where $t_1 < t_2$ means that t_1 precedes t_2. We assume the ordering $<$ is irreflexive, transitive, linear, and dense; also, in one place, we assume that the order is continuous (a second-order property).

The only axioms we assert here are that the start of an event precedes its end and that processes are active on open intervals:

(**AxOcc**). $Occurs(E, t_1, t_2) \rightarrow t_1 < t_2$
(**AxAct**). $Active(P, t) \rightarrow \exists t_1 t_2 (t_1 < t < t_2 \wedge \forall t' (t_1 < t' < t_2 \rightarrow Active(P, t')))$

We define a number of additional predicates, as follows:

An event-type is **discrete** if distinct occurrences cannot overlap:

$$Discrete(E) =_{\text{def}} \forall t_1 t_2 t_3 t_4 (Occurs(E, t_1, t_2) \wedge Occurs(E, t_3, t_4) \rightarrow t_2 \leq t_3 \vee t_4 \leq t_1 \vee (t_1 = t_3 \wedge t_2 = t_4)))$$

A process is **locally finite** if it neither always has been, nor always will be, active:

$$LocFin(P) =_{\text{def}} \forall t \exists t_1 t_2 (t_1 < t < t_2 \wedge \neg Active(P, t_1) \wedge \neg Active(P, t_2)).$$

Subtype: The relation $\sqsubseteq \subset (\mathcal{E} \times \mathcal{E}) \cup (\mathcal{P} \times \mathcal{P})$ is defined by

$$E_1 \sqsubseteq E_2 =_{\text{def}} \forall t_1 t_2 (Occurs(E_1, t_1, t_2) \rightarrow Occurs(E_2, t_1, t_2))$$
$$P_1 \sqsubseteq P_2 =_{\text{def}} \forall t (Active(P_1, t) \rightarrow Active(P_2, t).$$

Equality for event-types and processes: For $X \in \mathcal{E} \cup \mathcal{P}$,

$$X_1 = X_2 =_{\text{def}} X_1 \sqsubseteq X_2 \wedge X_2 \sqsubseteq X_1$$

Chunking: The function $chunk : \mathcal{P} \rightarrow \mathcal{E}$ is defined contextually, via an occurrence condition for the event-type $chunk(P)$, as follows:[12]

$$Occurs(chunk(P), t_1, t_2) =_{\text{def}} t_1 < t_2 \wedge \forall t (t_1 \leq t \leq t_2 \rightarrow (Active(P, t) \leftrightarrow t_1 < t < t_2))$$

Dechunking: The function $dechunk : \mathcal{E} \rightarrow \mathcal{P}$ is defined contextually, via an activity condition for the process $dechunk(E)$, as follows:[13]

$$Active(dechunk(E), t) =_{\text{def}} \exists t_1 t_2 (t_1 < t < t_2 \wedge Occurs(E, [t_1, t_2]))$$

Using these axioms and definitions we can prove:

Theorem 1. $Discrete(chunk(P))$.

Theorem 2. $Discrete(E) \rightarrow LocFin(dechunk(E))$.

[12] The first conjunct of the definiens is required to ensure that $chunk(P)$ satisfies **AxOcc**.

[13] The legitimacy of this definition depends on the fact, easily proved, that $dechunk(E)$, so defined, satisfies **AxAct**.

The converse of Theorem 2 does not hold: if E has only two occurrences, which overlap, then $dechunk(E)$ is locally finite but E is not discrete.

The next two theorems show that for discrete events and locally finite processes, *chunk* and *dechunk* are mutually inverse.

Theorem 3. $Discrete(E) \rightarrow chunk(dechunk(E)) = E.$

Theorem 4. $LocFin(P) \rightarrow dechunk(chunk(P)) = P.$

Note that even if P is not locally finite we have $dechunk(chunk(P)) \sqsubseteq P$, which holds for any process P.

We define two different flavours of sequential composition operator, which we call *weak* and *strong*. Other definitions are possible.

Weak Sequential Composition:

$$\begin{aligned} &Occurs(E_1; E_2, t_1, t_2) \ =_{\text{def}} \\ &\quad \exists t_3 t_4 (t_1 < t_3 \leq t_4 < t_2 \wedge Occurs(E_1, t_1, t_3) \wedge Occurs(E_2, t_4, t_2)) \end{aligned}$$

The next theorem establishes the associativity of weak sequential composition:

Theorem 5. $E_1; (E_2; E_3) = (E_1; E_2); E_3.$

As a result of this theorem, we can drop the parentheses and write $E_1; E_2; E_3$. As noted in the main text, under Weak Sequential Composition $E_1; E_2$ is not discrete.

Strong Sequential Composition:

$$\begin{aligned} Occurs(E_1 \hat{;} E_2, t_1, t_2) =_{\text{def}} \exists t_3 t_4 (&t_1 < t_3 \leq t_4 < t_2 \\ &\wedge Occurs(E_1, t_1, t_3) \wedge Occurs(E_2, t_4, t_2) \\ &\wedge \neg \exists t, t' ((t_3 \leq t < t_2 \wedge Occurs(E_1, t, t')) \\ &\qquad \vee (t_1 < t' \leq t_4 \wedge Occurs(E_2, t, t')))) \end{aligned}$$

The next theorem establishes that the strong sequential composition of two discrete events is discrete.

Theorem 6. $Discrete(E_1) \wedge Discrete(E_2) \rightarrow Discrete(E_1 \hat{;} E_2).$

As noted in the main text, the operator $\hat{;}$ is not associative.

For **repetition**, we want to define a process $rep(E)$ which is active during a period in which E repeatedly occurs. The simplest case is where E occurs twice. This can be expressed as an occurrence of the event $E; E$ (but not $E \hat{;} E$, since $E \hat{;} E$ cannot occur). We define:

$$rep(E) =_{\text{def}} dechunk(E; E)$$

This would mean that two occurrences of E suffice for this process to be active. Normally we would expect a larger number (think of our bursts of machine-gun fire). We could arbitrarily decide for some n that we require $rep(E) = dechunk(E; E; \cdots; E)$, where the right-hand side contains n copies of 'E'. While it would clearly not be feasible to fix an n which will always give satisfactory results, the important thing is that as we increase n we obtain a sequence of processes each of which is special case of the previous one. This is shown by the following theorem:

Theorem 7. $dechunk(E;E;E) \sqsubseteq dechunk(E;E)$.

As well as the indeterminacy as to how many repetitions of E are required before we say that the process $rep(E)$ is active, there is an indeterminacy as to how far apart the individual occurrences of E must be in time. Resolution of both these indeterminacies must depend on the nature of the specific event-type in question and the context in which it is considered.

References

1. Abler, R., Adams, J.S., Gould, P.: Spatial Organization: The Geographer's View of the World. Prentice-Hall International, Englewood Cliffs (1971)
2. Adaikkalavan, R., Chakravarthy, S.: SnoopIB: interval-based event specification and detection for active databases. In: Kalinichenko, L.A., Manthey, R., Thalheim, B., Wloka, U. (eds.) ADBIS 2003. LNCS, vol. 2798, pp. 190–204. Springer, Heidelberg (2003)
3. Aitken, S., Curtis, J.: Design of a process ontology: Vocabulary, semantics, and usage. In: Gómez-Pérez, A. (ed.) Proceedings of the 13th International Conference on Knowledge Engineering and Knowledge Management (EKAW02), pp. 108–113 (2002)
4. Batty, M.: Geocomputation using cellular automata. In: Openshaw, S., Abrahart, R.J. (eds.) GeoComputation, pp. 95–126. Taylor and Francis, London (2000)
5. Bivand, R., Lucas, A.: Integrating models and geographical information systems. In: Openshaw, S., Abrahart, R.J. (eds.) GeoComputation, pp. 331–363. Taylor and Francis, London (2000)
6. Bregt, A.K., Bulens, J.: Integrating GIS and process models for land resource planning. In: Heineke, H., et al. (eds.) European Soil Bureau Research Report No. 4, pp. 293–304. Laboratory of GeoInformation Science and Remote Sensing, Wageningen University, The Netherlands (1998)
7. Brown, D.G., Riolo, R., Robinson, D.T., North, M., Rand, W.: Spatial process and data models: towards integration of agent-based models and GIS. J. Geogr. Syst. **7**, 25–47 (2005)
8. Claramunt, C., Parent, C., Thériault, M.: Design patterns for spatio-temporal processes. In: Spaccapietra, S., Maryanski, F. (eds.) Searching for Semantics: Data Mining, Reverse Engineering, pp. 415–428. Chapman and Hall, New York (1997)
9. Couclelis, H.: Cellular worlds: a framework for modeling micro-macro dynamics. Environ. Plann. A **17**, 585–596 (1985)
10. Reis Ferreira, K., Camara, G., Monteiro, A.M.V.: An algebra for spatiotemporal data: from observations to events. Trans. GIS **18**(2), 253–269 (2014)
11. Galton, A.: The Logic of Aspect: An Axiomatic Approach. Clarendon Press, Oxford (1984)
12. Galton, A.: Dynamic collectives and their collective dynamics. In: Cohn, A.G., Mark, D.M. (eds.) COSIT 2005. LNCS, vol. 3693, pp. 300–315. Springer, Heidelberg (2005)
13. Galton, A.: Eventualities. In: Fisher, M., Gabbay, D., Vila, L. (eds.) Handbook of Temporal Reasoning in Artificial Intelligence, pp. 25–58. Elsevier, New York (2005)
14. Galton, A.: Experience and history: processes and their relation to events. J. Logic Comput. **18**, 323–340 (2008)

15. Galton, A.: States, process and events, and the ontology of causal relations. In: Donnelly, M., Guizzardi, G. (eds.) Formal Ontology in Information Systems: Proceedings of the 7th International Conference (FOIS 2012), pp. 279–292 (2012)
16. Galton, A., Worboys, M.: Processes and events in dynamic geo-networks. In: Rodríguez, M.A., Cruz, I., Levashkin, S., Egenhofer, M. (eds.) GeoS 2005. LNCS, vol. 3799, pp. 45–59. Springer, Heidelberg (2005)
17. Gehani, N.H., Jagadish, H.V., Shmueli, O.: Event specification in an active object-oriented database. ACM SIGMOD Rec. **21**(2), 81–90 (1992)
18. Goodchild, M.F., Yuan, M., Cova, T.J.: Towards a general theory of geographic representation in GIS. Int. J. Geogr. Inf. Sci. **21**(3), 239–260 (2007)
19. Harel, D.: Dynamic logic. In: Gabbay, D., Guenthner, F. (eds.) Handbook of Philosophical Logic. volume II: Extensions of Classical Logic, pp. 497–604. Reidel, Dordrecht (1984)
20. Hazelton, N.W.J., Leahy, F.J., Williamson, I.P.: Integrating dynamic modeling and geographic information systems. URISA J. **4**(2), 47–58 (1992)
21. Langran, G., Chrisman, N.R.: A framework for temporal geographic information. Cartographica **25**(3), 1–14 (1988)
22. Moens, M., Steedman, M.: Temporal ontology and temporal reference. Comput. Linguist. **14**, 15–28 (1988)
23. O'Sullivan, D., Unwin, D.J.: Geographic Information Analysis. Wiley, Hoboken (2003)
24. Peuquet, D.J.: It's about time: a conceptual framework for the representation of temporal dynamics in geographic information systems. Ann. Assoc. Am. Geogr. **84**, 441–461 (1994)
25. Peuquet, D.J., Duan, N.: An event-based spatiotemporal data model (ESTDM) for temporal analysis of geographical data. Int. J. Geogr. Inf. Syst. **9**(1), 7–24 (1995)
26. Raper, J., Livingstone, D.: Development of a geomorphological data model using object-oriented design. Int. J. Geogr. Inf. Syst. **9**, 359–383 (1995)
27. Shank, R.C., Abelson, R.: Scripts, plans, goals and understanding. Erlbaum, Hillsdale (1977)
28. Snodgrass, R.T.: Temporal databases. In: Frank, A.U., Formentini, U., Campari, I. (eds.) GIS 1992. LNCS, vol. 639, pp. 22–64. Springer, Heidelberg (1992)
29. Takeyama, M., Couclelis, H.: Map dynamics: integrating cellular automata and GIS through geo-algebra. Int. J. Geogr. Inf. Sci. **11**, 73–91 (1997)
30. Tobler, W.R.: Cellular geography. In: Gale, S., Olsson, G. (eds.) Philosophy in Geography, pp. 379–386. D. Reidel, Dordrecht (1979)
31. Torrens, P.M.: Process models and next-generation geographic information technology. ArcNews Online, Summer 2009. http://www.esri.com/news/arcnews/summer09articles/process-models.html
32. Torrens, P.M., Benenson, I.: Geographic automata systems. Int. J. Geogr. Inf. Sci. **19**(4), 385–412 (2005)
33. Worboys, M.: Event-oriented approaches to geographic phenomena. Int. J. Geogr. Inf. Sci. **19**, 1–28 (2005)
34. Worboys, M.F., Hornsby, K.: From objects to events: GEM, the geospatial event model. In: Egenhofer, M., Freksa, C., Miller, H.J. (eds.) GIScience 2004. LNCS, vol. 3234, pp. 327–343. Springer, Heidelberg (2004)
35. Yuan, M.: Representing complex geographic phenomena in GIS. Cartography Geogr. Inf. Sci. **28**(2), 83–96 (2001)