Events, Neural Systems and Time Series

Leslie Smith[1], Daniel Metz[2], Jungpen Bao[3], and Pedro Bizarro[4]

[1] Computing Science and Mathematics, School of Natural Sciences, University of Stirling, Stirling FK9 4LA, UK
[2] Information Systems Institute, University of Siegen, Hölderlinstraße 3, 57076 Siegen, Germany
[3] Department of Computer Science and Technology, Xi'an Jiaotong University, Xi'an 710049, China
[4] Informatics Engineering Department, University of Coimbra, 3030-290 Coimbra, Portugal

Abstract. Different types of events occurring in computer, neural, business, and environmental systems are discussed. Though events in these different domains do differ, there are also important commonalities. We discuss the issues arising from automating complex event handling systems.

Keywords: complex events, autonomous robotics, computing events, cognitive events

1 Introduction: events in different contexts

Events can be considered as a unifying paradigm crossing disciplinary boundaries. Events happen[1]: events are considered to occur at a particular time and place, and to involve a number of participant entities. Generally, events have (a non-null set of) causes, and also (a non-null set of) effects[2]. Events happen both inside a system and in its environment. Events occur in all types of area: here we discuss events in computer systems and networks, neural systems, business, and the environment, but this represents only four example areas. What we consider to be an event often depends on our standpoint, and what we are currently interested in: events that are not of interest still occur, but we ignore them.

In terms of computer systems and networks, events range from the change of state of a signal line inside a CPU chip or the failure to find a datum in a CPU cache, to the pressing of a button on a mouse, to receipt of a protocol packet, to failure of a link, or, at a higher level, the placing of an order or the receipt of a shipment, or at a still higher level the introduction of a new device to the market or the release of a new web portal. There are a number of different theoretical bases that can and have been used for events primarily in this context. These range from Petri nets [12] to calculus of communicating systems

[1] "Events dear boy, events": ascribed to Harold Macmillan, ex Prime Minister of the UK, when asked by a reporter what was most likely to blow a government off course.
[2] Events with no effects can safely be ignored

[9] to communicating sequential processes [6] to communicating agents [10]. Each of these has been taken up in some areas of computer systems and networks, but they have not yet found application outside of this domain. Here, we are concerned with what the events are, rather than with analysing or organising them. Clearly these different levels of events are inter-related: each higher-level event actually is made up of a large number of lower level events.

In terms of neural systems, one needs to start by considering what the events might be[3]. Events could range from arrival of an action potential at a synapse (one might start at a lower level, for example with the release of a vesicle from the presynaptic terminal, or the arrival of molecules of neurotransmitter from such a vesicle at a single post-synaptic ion channel), or with the detection of some percept in the auditory or visual cortex, or the execution of a motor command. There is little agreement about the nature of information transfer inside the brain (beyond that it is mediated by action potentials), which makes the issue of what is an event controversial (particularly at the level above single spikes: is an event the reception of a stream of spikes from one other neuron? Or a volley of spikes from a number of neurons? Or the collection of spikes that make up a synfire chain?)

In terms of businesses, events may take many forms, ranging from the initiation of the development of a new product, the release of a new product (by that particular business, or one of its competitors), to changes in prices of raw materials, or the opening of a new factory or office. There is a standardised notation for these events developed by the Object Management Group/Business Process Management Initiative (`http://www.bpmn.org/`): see [13]: this is oriented towards orchestration of these events in a computational context. As with computer events, many of these can be decomposed: for example the opening of a new factory is the culmination of a long sequence of events which presumably started with a decision being taken to open a new factory (or rather, before that, with the events that led to the requirement for a new factory). The relevant events may be internal to the business, or may be within the business's operating environment.

In terms of events in the (natural) environment, these may be associated with a particular occurrence, for example a tree falling over in a forest. As for both computer events and neural events, this event is made up from many other events: in this case it can be considered as the culmination of a set of less visible events that started, perhaps, with the seed from which the tree grew sprouting. We note also that the environment may refer to the environment of some other entity (such as a computer, or a business) in which case the events of interest will be those affecting the that entity.

[3] It may be that events are not the best way of describing what is happening in such systems, but they are nonetheless useful, and give us a means of comparing different system types

The primary difference between computer events[4] and the neural and environmental events is that for the computer events, we know how lower level events lead to higher level events, and how higher level events are orchestrated by lower level events, whereas for both the environmental and neural systems this knowledge is a great deal patchier. In systems which have been designed, the event hierarchy is part of that design. Unlike built systems, natural systems do not need to keep to a careful hierarchy of events they can (and surely do) cross putative levels: indeed, one could argue that the association of levels with events is a human way of organising these otherwise unorganised events. Events in the business area lie somewhere between these poles: they often will have a clear set of constituent (sub-)events. However, there will also be events with a much less clearly defined hierarchy of (sub-)events (for example, a key employee leaving).

The issue then is whether this event-based view of business, environmental and neural systems is useful. It clearly is useful for computer systems and networks. For business systems, there can be a direct connection to events in computer systems, although this does not capture all the events in a business context. For neural systems, we currently always need to ask about the level of the events in which we are interested: further and most critically, we do not understand how synaptic/action potential events lead to higher level events. (We have some understanding of how such events lead to further synaptic/action potential events, but these further events are essentially the same level. Are there higher level events in neural systems? We clearly do have (first-person) mental events, and we believe that these are mediated by action potential/synaptic events: but the relationship between these two is not yet anything like fully elucidated. There are higher-level theories of mental function (such as those of [5] or [3]), but these are largely narrative models[5] rather than precise simulations. For natural environmental systems, the issues are perhaps easier, since we already have a large volume of physics, much of which is about the inter-relationship between events in the physical world. The events that we are interested in can only be those that we can detect or infer, and these are then the inputs to our computational systems (and, indeed, our own neural systems as well).

2 Contexts 1: events and robotics

Robots interact with their environment and this interaction can be considered to be mediated by events. These may be generated by the environment, and sensed by way of the robots sensors (whether visual, auditory, tactile, or whatever), or they may be generated by the robot, in which case we would more usually call them actions. These output events are not sensed as such by the environment, but result in the generation of new and perhaps different events by the environment as detected by the robots sensors. There may also be events generated by the

---

[4] In complex event processing (CEP) an event is an object that can be subjected to computer processing [8]: but this is a somewhat circular definition here.

[5] That is, stories of how a system might work

robot which are internal: what we might describe as cognitive events, resulting from sensory events, and perhaps actions as well.

Real (natural) environments are not well defined[6]. Environments are inherently changeable, and largely unpredictable. Even simple actions, such as turning a robots driving wheel will not always have the same effect wheels slip, for example. Further, the sensor data received from the environment is always uncertain: this may be due to noise, or deficiencies in the sensors themselves, but may also be due to variation in the environment, for example variations in the lighting affecting the visual sensor, or extraneous sounds and reflections affecting an auditory sensor.

The aim in much of robotics is effective autonomous operation, in spite of these difficulties. Unpredictability and variation in operation are just some of the problems in this area: there are others as well, such as goal setting, and adjudging performance.

3 Contexts 2: Time series

Time series are sequences of data, often measurements, over a period of time. Each value can be considered to be an event. Often the values are recorded at regular intervals, but this need not always be the case. Analysis of time series may have several aims: classification of the time series, prediction of future events (such as the next few values of the time series), or error or novelty detection (that is: has something generating the time series changed in some way).

Time series arise in many domains (certainly including all those discussed above). One particular area of commercial interest is in demand forecasting. Utilities (gas, electricity, water etc.), automated teller machine networks, telephone companies, internet service providers, call centres and many other businesses all have multiple levels of seasonal variation in demand. The operators of these industries want to know (for example) likely immediate and short-term requirements, as well as if some particular form of error condition (for example large scale leaks in the water industry, or system failures in an ATM network) holds. Further, the environment in which these time series values are being recorded is not constant: external events can have a major effect on the time series. For example, major sporting events, of sudden changes in weather, or even television programming may influence user behaviour in a non-random way, and thus result in major changes in demand.

The operators have good reasons for needing this information: altering their capacity to respond to alterations in demand may require time. For utilities, generators may need to be started up, power grid lines or pipelines reassigned, or (for ISPs) new servers assigned, or low-bandwidth pages set up. These operators have had coped with these problems for many years: they have staff whose job it is to predict demand, and their experience is a major asset to these companies.

[6] This might, of course, be different for an artificial environment, such as that in a game or other simulation.

4 Automating complex event handling systems

In both the above contexts, there are existing solutions. But these solutions are incomplete. Robotic behaviour is generally brittle, automated time series prediction is known to be a difficult problem, and understanding the ways in which human predictors of demand operate (or eliciting their knowledge) can be difficult. Static rule based systems work only in static known environments. This suggests that adaptive systems will be necessary to cope with altering environments. But how should such adaptation be implemented? When should a system learn? What should a system learn? How should what is learned be applied to the problem at hand?

In the relatively restricted case of time series prediction, at first sight, these questions appear to boil down to issues of selection of mechanisms for prediction (of which there are many). However, in reality, there is extraneous information available as well as the actual numbers in the time series. Then one needs to consider which aspects of this additional available information should be used or even if one should be seeking out further information not currently available. In the relatively less restricted domain of autonomous robotics, these two aspects of what might be learned become issues of altering the decision system, and seeking out appropriate perceptual input (that is, issues of active perception). The robot can move and/or alter its sensors to alter what might be detected by its sensors. Further, by appropriate movement, it can learn more about what it is sensing, because it knows what it has altered in order to alter the sensation received. Animals do this all the time, moving their whole bodies or their heads to alter visual, auditory and olfactory perceptions. In a business context, finding appropriate extraneous information, and bringing it to the locations at which decisions are made is a difficult task: one can argue that neural decision making shares many of the same problems, particularly relating to the range of possibly relevant information that might be available. The amount of data that might be processed is generally huge: as we attempt to process more, the problem becomes harder, but if we restrict what is processed, the lack of data may mean that important information is lost. Knowing or learning what matters, and what may safely be ignored is is challenging: a related problem exists in statistics where it is called feature selection [2][7], and remains the subject of current research.

Non-static situations require non-static responses: we cannot expect to predetermine all responses or predictions. Such learning needs to be underpinned by some form of change or adaptation. Widder et al. present an approach to identify suspicious, unknown events in an event cloud [15]. Discriminant analysis is applied to detect unknown or suspicious combinations of events which havent be seen in the past. This approach can be used for fraud detection (see the ATM example above). We believe it is possible to make a learning system more robust than a non-learning system, but such an outcome is not necessarily the case. Simple neurally inspired learning systems (such as back-propagating neural networks or radial basis function networks) essentially learn statistical information about their environments [1]. For such learning systems, the larger the volume of training data available, the better: although even here, the data needs to be appropriate and clean: feature selection is again important. There are other types of learning as well, for example reinforcement learning [14], where what is required is a signal to show when the behaviour is appropriate (reward) or inappropriate (punishment). In the case of event streams, similarity search techniques are applied to react to recurrent situations and to predict (business) processes (e.g.,[11]). Similarly, historical data (events) can be analyzed to establish a proactive control of manufacturing processes (e.g., [4]). The similarity search techniques are better, if the volume of available data (events) increases. Nevertheless, a risk might be the solidification of what has been learned in the past (that is, innovation can be hindered). But even for animals, learning can go wrong, as appears to be the case in, for example, autism.

One advantage that modern computer systems have in this area is the availability of large amounts of processing power and memory. Parallelism means that, for example, all the sensors in a robotic system can be processed at the same time, and cross-modal percepts can be computed, continuously. Thus even if certain types of processed sensor information are not always required, they can be made instantaneously available. Further, memory can be used to look back on actions taken and predictions made in the light of more recent events, and these can, for example, be re-run using different learning techniques to adjust what might be applied to future events, enabling more sophisticated learning techniques. The issue is determining which particular current and historical events are most relevant. Similarity search techniques may also prove useful in this context.

5 Conclusions

Events do provide a useful unifying paradigm across a wide range of domains, including the four that we have discussed here. We can interpret events both in terms of the input (from outside) to a system, and the output (to outside) from a system, as well as being internal to either the system itself, or the environment of the system. This is useful in the making the system/environment concept applicable in areas outside of the robot/environment system area, for example, in the business/business environment domain. A similar approach can be taken to the application of event-based approaches to prediction/error-detection in utilities, and these can use the same types of learning based approaches, and event selection techniques.

There are some differences: e.g. the nature of active sensing is different in utilities, (where it is about the search for appropriate extraneous information) from in robotics (where it is about the way in which the robot itself can alter its own input), and business events come in many diverse forms, not all of which can easily be placed in formal contexts such as those in [13]. However, events can be used in quite different domains, and similar techniques and technologies used across those domains. There remains, however, a need to work on the best way of organising and theorising about these events, and in selecting a technology for this which can be used by those designing and analysing these systems across different domains.

References

1. Bishop, C.: Neural Networks for Pattern Recognition. Oxford University Press (1995)
2. Blum, A., Langley, P.: Selection of relevant features and examples in machine learning. Artificial intelligence 97(102), 245–271 (1997)
3. Garforth, J., McHale, S.L., Meehan, A.: Executive attention, task selection and attention-based learning in a neurally controlled simulated robot. Neurocomputing 69(16-18), 1923–1945 (2006)
4. Grauer, M., Karadgi, S., Müller, U., Metz, D., Schäfer, W.: Proactive control of manufacturing processes using historical data. Lecture Notes in Computer Science 6277, 399–408 (2010)
5. Hawkins, J., Blakeslee, S.: On Intelligence. Times Books (2004)
6. Hoare, C.: Communicating Sequential Processes. Prentice-Hall (1986)
7. Liu, H., Motoda, H., Setiono, R., Zhao, Z., Chawla, S., Salehi, E., Nyayachavadi, J., Gras, R., Zagoruiko, N., Borisova, I.: Feature selection in data mining. In: JMLR Workshop and Conference Proceedings. vol. 10 (2010)
8. Luckham, D.: The Power of Events. Addison-Wesley (2007)
9. Milner, R.: A Calculus of Communicating Systems, vol. 92. Springer (1980)
10. Milner, R.: The Space and Motion of Communicating Agents. Cambridge (2009)
11. Obweger, H., Suntinger, M., Schiefer, J., Raidl, G.: Similarity searching in sequences of complex events. In: Proc. of Int. Conf. On Research Challenges in Information Science (RCIS). pp. 631–639 (2010)
12. Petri, C.: Communication with automata. Tech. Rep. AD0630125, DTIC (1966)
13. Shapiro, R., White, S., Palmer, N., zur Muehlen, M., Allweyer, T., Gagne, D., Silver, B., Fischer, L.: BPMN 2.0 Handbook. Future Strategies Inc. (2010)
14. Sutton, R., Barto, A.: Reinforcement Learning: An introduction. MIT Press (1998)
15. Widder, A., von Ammon, R., Schaeffer, R., Wolff, C.: Identification of suspicious, unknown event patterns in an event cloud. In: Jacobsen, H.A., Mühl, G., Jaeger, M.A. (eds.) Proceedings of the 2007 Inaugural International Conference on Distributed Event-Based Systems, DEBS 2007. pp. 164–170 (2007)