Introduction to Reasoning in Event-Based Distributed Systems

Sven Helmer, Alex Poulovassilis, and Fatos Xhafa

1 Event-Based Distributed Systems

Event-based distributed systems have played an important role in distributed computing in the recent past and their influence is steadily increasing. With the rapid development of Internet technologies, such systems are gaining in importance in a broad range of application domains, including enterprise management, environmental monitoring, information dissemination, finance, pervasive systems, autonomic computing, geo-spatial systems, collaborative working and learning, and online games. Event-based computing is becoming a central aspect of emerging large-scale distributed computing paradigms such as Grid, Peer-to-Peer and Cloud computing, wireless networking systems, and mobile information systems.

The general motivation for distributed processing (which applies also to event-based distributed systems) is that it allows for more scalable and more reliable systems. Moreover, in the context of event-based distributed systems specifically, events occurring over a wide area can be partially filtered and aggregated locally before being shipped to their destination, i.e. to the component or components of

Sven Helmer
Department of Computer Science and Information Systems, Birkbeck,
University of London, UK
Tel.: +44 (0)20 7631 6718; Fax: +44 (0)20 7631 6727
e-mail: `sven@dcs.bbk.ac.uk`

Alex Poulovassilis
London Knowledge Lab, Birkbeck, University of London
Tel.: +44 (0)20 7631 6705 / (0)20 7763 2120; Fax: +44 (0)20 7631 6727
e-mail: `ap@dcs.bbk.ac.uk`

Fatos Xhafa
Department of Languages and Informatic Systems, Technical University of Catalonia, Spain
Tel.: +34 93 4137880; Fax: +34 93 4137833
e-mail: `fatos@lsi.upc.edu`

the system responsible for processing information about event occurrences and reacting appropriately.

In event-based systems, an ‘event’ can take many forms. However, common to all events is that they are abstractions of observations made in a certain environment. Event-based processing originated in the area of active databases in the 1980s with the introduction of triggers to database systems (we refer readers to [9, 11] for an overview of active databases). A trigger is an Event-Condition-Action (ECA) rule which is checked by the DBMS whenever an event occurs that matches the Event part of the rule. The Condition part of the rule acts as a sentinel filtering out events that are not relevant, and only those events satisfying the condition cause the Action part of the rule to be executed. Since these early beginnings, ECA rules have become a reactive computing paradigm in their own right, independent of their use in database systems. For example, the ECA concept has been applied in the context of middleware architectures such as CORBA [6], J2EE [2] and TIBCO [4], in stream processing [5], reactive web-based applications [8], wireless sensor networks [1] and radio frequency identification [10]. We refer readers to [3] for an overview of the development of the ECA paradigm in databases during the 1980s and 1990s, and a discussion of its more recent use in areas such as distributed event specification and detection, information filtering, web page change monitoring, stream data processing, role-based access control and autonomic computing.

What are some of the reasons for the popularity of event-based processing? Firstly, it offers a lot of flexibility to applications that need to monitor an environment and react to the occurrence of significant events. Event-based systems rely mainly on asynchronous communication to do this. In contrast to synchronous communication, processes do not have to stop working while waiting for data. Generally this means that event sources ‘push’ data about event occurrences to event consumers. Moreover, developers are able to build more loosely-coupled systems, as system components only need minimum knowledge about each other. Different levels of coupling are possible, ranging from fixed point-to-point communications to a network of brokers.

Event-based processing began with systems that handled simple events, such as the insertion of a tuple into a database table or the reading of a temperature value by a sensor. However, as systems are processing increasing volumes of events, users are not necessarily interested in keeping track of thousands or even millions of simple events, but rather in seeing and handling the ‘big picture’. This can be achieved by employing intelligent techniques for deriving events at higher levels of abstraction. For example, the area of Complex Event Processing (CEP), which is the focus of several of the chapters in this book, aims to correlate simple events temporally and/or spatially in order to infer more meaningful composite events. CEP comes with a trade-off, though, in that a balance needs to be struck between the expressiveness of CEP languages on the one hand and the performance of CEP systems processing them on the other: the more expressive a language, the more powerful the event processing engine needs to be.

Many different architectures, languages and technologies have been and are being used for implementing event-based distributed systems — we refer readers to [7] for an overview of the range of technologies used and an analysis of common research issues arising across different application domains. This large variability is due to the fact that much of the development of such systems has been undertaken independently by different communities. While there is some overlap in the different approaches, often the main driving force during development has been the needs and requirements of specific application domains. Although a merging and generalisation of concepts is under way, c.f. [7], this will still take some time to accomplish. In the meantime, Chapters 2 and 3 of this book present an overview of distributed architectures for event-based systems (Chapter 2) and languages for specifying and detecting complex events (Chapter 3), aiming to allow the reader to gain an understanding of the state-of-the-art in these two fundamental areas of the field.

Many research challenges are being addressed by the event-based distributed computing research community, ranging from the detection of events of interest, to filtering, aggregating, dissemination and querying of event data, and more advanced topics relating to the semantics of event languages and issues such as performance and optimisation of event processing, specification and implementation of event consumption policies (i.e. policies for consuming simple events during the derivation of composite events), handling event data arising from heterogeneous sources, resource management, ensuring the privacy and security of information, and quality of service guarantees. Intelligent techniques are playing an important role in addressing these challenges. In particular, reasoning and logic-based approaches provide fundamental principles for tackling these research issues and developing solutions on the basis of formal, yet powerful, foundations. Chapters 4 to 9 of the book describe a broad range of current research in this direction. Chapters 12 and 13 focus on two specialist topics, namely the role of context in event-based distributed systems, and the handling of uncertainty in the detection of simple events and the inference of higher-level derived events from simple events.

Event-based systems can serve as a basis for the development of advanced applications in diverse domains, ranging from traditional areas of logistics and production to more recent mobile and pervasive applications. As event-based processing is often application-driven, the concepts presented in the book are illustrated using examples from a broad range of application scenarios in areas such as supply chain management, environmental and traffic monitoring, patient monitoring, data centre and network monitoring, fraud detection, smart homes, access control, spacecraft and satellite data monitoring, online collaboration, monitoring market data, and monitoring business processes. Chapters 9, 10 and 11 focus specifically on applications in event-based pervasive computing, patient care, and collaboration in social organisations.

2 Reasoning in Event-Based Distributed Systems

Reasoning about the properties and behaviour of event-based distributed systems differs in many ways from reasoning about traditional event management systems due to their greater complexity and dynamicity as well as their temporal, spatial, domain-specific and context-sensitive characteristics. A variety of different event types may arise in event-based distributed systems, including signal, textual, visual and other kinds of events. The time associated with an event may be a single time point or a time interval. Reasoning in event-based distributed systems thus needs to address new issues and research challenges including:

- correlating simple distributed events and reasoning about them in real-time;
- spatio-temporal reasoning over events, with both point and interval semantics;
- reasoning about uncertain events and under real-time constraints;
- reasoning about events in continuous time;
- context-aware reasoning.

Depending on the application supported by an event-based system, different architectures and models are being used, including publish-subscribe, Peer-to-Peer (P2P), Grid computing, event-stream processing, and message queues. A common factor across different application domains is an ever-increasing complexity. Users and developers expect that modern systems are able to cope with not only simple events reporting a change in a single data item but also composite events, i.e. the detection of complex patterns of event occurrences that are possibly spatially distributed and may span significant periods of time. This gives rise to the need for development of

- semantic foundations for event-based distributed systems;
- formal models and languages for expressing composite events;
- reasoning techniques for such models and languages;
- formal foundations for event consumption policies;
- data mining and machine learning techniques for detecting new or unusual event patterns;
- scalable techniques for processing efficiently large volumes of complex distributed events.

The event-based approach is becoming a central aspect of new Grid and P2P systems, ubiquitous and pervasive systems, wireless networking, and mobile information systems. The architectures and middleware of such systems use event-based approaches not only to support efficiency, scalability and reliability but also to support a new class of demanding applications in a variety of domains. Event-based approaches are showing their usefulness not only for stand-alone platforms and applications but also for achieving interoperability between, and integration of, different event-based applications, particularly in the business domain. Research challenges here include:

- development of distributed architectures for supporting intelligent event processing;
- development of event-driven Service Oriented Architectures to support interoperability and integration of distributed applications;
- handling the heterogeneity that arises when event data is produced by different sources, e.g. by enriching the data with additional semantic metadata or through ontology-based mediation services;

- developing techniques for specifying, analysing and enforcing policies for resource management, security and privacy;
- monitoring and delivering quality of service requirements.

Further research issues are identified in Chapter 2 of this book, which gives an overview of the architectural aspects of event-based distributed systems, focusing particularly on the intelligent and reasoning techniques incorporated within such systems. Some of these are explored in more detail in later chapters of the book, including:

- optimisation of event processing in the face of large numbers of distributed users, high volumes of distributed event data, and the need for timely response with low resource consumption;
- dynamic adaptation to new situations as applications are executing, e.g. changes in the availability of resources, the types of event data being produced, the complex events that need to be detected, or the quality of service requirements;
- specifying, implementing and reasoning with policies for security and access control to physical and virtual environments and resources;
- responsiveness to the requirements of diverse users, for example through rule-based mechanisms for capturing different users needs and preferences and making recommendations to users.

3 Overview of the Book

This book aims to give a comprehensive overview of recent advances in intelligent techniques and reasoning in event-based distributed systems. It divides roughly into five sections:

(i) Chapters 2 and 3 survey different architectures for event-based distributed systems and different languages for complex event processing, covering foundational material that is developed further in subsequent chapters of the book.
(ii) Chapters 4, 5 and 6 explore in more detail some of the event language paradigms identified in Chapter 3, addressing issues such as semantics, performance and optimisation of event-based processing.
(iii) Chapters 7 and 8 focus on security and access control in event-based distributed systems.
(iv) Chapters 9, 10 and 11 discuss the requirements for intelligent processing in several application domains and the development of techniques targeting these domains: event detection in pervasive computing environments, emergency patient care, and online collaboration in social organisations.
(v) Chapters 12 and 13 conclude by addressing two specialist aspects of reasoning in event-based distributed systems: handling of context and handling of uncertainty.

Overview of the Chapters

Chapter 2, *“Distributed architectures for event-based systems”*, gives an overview of the architectural aspects of event-based distributed systems. The authors present a logical architecture that comprises the main components involved in generating, transmitting, processing and consuming event data, based on the ECA paradigm. They discuss the nature of primitive, composite and derived events, and the ways in which these are processed in this logical architecture. They then describe how to realise the archetypal logical architecture in a variety of settings. The alternative system architectures they present are discussed within five themes — Complex Event Processing, Service Oriented Architectures, Grid Architectures, P2P Architectures and Agent-based Architectures. For each of these themes, the authors discuss previous work, recent advances, and future research trends.

Chapter 3, *“A CEP Babelfish: Languages for Complex Event Processing and Querying Surveyed”*, describes five commonly used approaches for specifying and detecting event patterns: (i) event query languages based on a set of event composition operators, (ii) data stream query languages, (iii) production rules, (iv) state machines, and (v) logic languages. The authors illustrate each of these approaches by means of a use case in Sensor Networks and discuss appropriate application areas for each approach. They note that there is no single best approach that fits all application requirements and settings, and that therefore commercial CEP products tend to adopt multiple approaches within one system or even combined within one language.

The next three chapters explore some of the language approaches (i)-(v) above in more detail. Chapter 4, *“Two Semantics for CEP, no Double Talk: Complex Event Relational Algebra (CERA) and its Application to XChange^{EQ}”*, discusses the semantics of Event Query Languages, from both declarative and operational perspectives. The authors note that for such languages the declarative semantics serve as a reference for a correct operational semantics, and also as the basis for developing optimisation techniques for event query evaluation. The chapter focuses particularly on the XChange^{EQ} event query language, which is based on the event composition operators approach. The authors adopt a model-theoretic approach for the declarative semantics, and an event relational algebra as the basis for the operational semantics, and hence for event query evaluation and optimisation. They confirm the relationship between the two versions of the semantics by sketching a proof of the correctness of the operational semantics with respect to the declarative semantics.

Chapter 5, *“ETALIS: Rule-Based Reasoning in Event Processing”*, presents a logic language for specifying complex events. Similarly to Chapter 4, the authors give both a model-theoretic declarative semantics for their language and a rule-based operational semantics. There is a detailed discussion of event consumption policies, defined over both time points and time intervals. Their language is compiled into Prolog for execution, and the authors compare the event detection performance of two Prolog implementations of their approach with a state machine-based implementation. A brief discussion follows of how a logic-based approach can support reasoning over complex events, for example as relating to their relative order and other more complex relationships between events.

Chapter 6, *"GINSENG Data Processing Framework"*, describes a hybrid approach to distributed complex event processing, combining the capabilities and benefits of publish/subscribe, event stream processing (ESP) and business rule processing (BRP). The authors describe the GINSENG modular middleware architecture for distributed event processing, which supports the integration of a variety of event producing and event consuming external components through its extensible content-based publish/subscribe mechanism. The middleware handles the stateless parts of business rules close to the data sources through the publish/subscribe mechanism, while the stateful parts of rules are handled by the ESP and BRP engines. The authors describe the implementation of these two engines, and focuses particularly on performance monitoring and control. They discuss data quality-driven optimisation of event stream processing, using an evolutionary approach. They also mention the possibility of using temporal reasoning to identify event occurrences that can no longer match any rule and can therefore be garbage-collected. The chapter concludes with a review of related work in middleware technology, publish/subscribe, ESP and BRP.

Chapter 7, *"Security Policy and Information Sharing in Distributed Event-Based Systems"*, discusses security policies and information sharing in event-based distributed systems, using health care service provision as the motivating application. The authors assume a publish/subscribe mechanism for event-based communication, and they discuss how role-based access control can be used to enforce authorisation policies at the client level, while taking context into account as well. They discuss additional requirements for providing secure information flow between distributed system components, both within and between administrative domains, and how these requirements can be implemented. They describe how reasoning techniques can be applied to formal policy specifications, so as to infer information about the flow of data within the system and the confidentiality and integrity properties of the system.

Chapter 8, *"Generalization of Events and Rules to Support Advanced Applications"*, proposes extensions to ECA rule syntax and semantics in order to support the requirements of advanced applications, exemplifying these requirements with examples drawn from the specification and implementation of policies for access control to physical spaces. The ECA rule extensions include: specification of alternative actions to be undertaken when the event part of a rule is detected but the condition is false; generalisation of event specification and detection using constraints expressed on the attributes of events; and detection of partial and failed complex events. The chapter discusses event detection techniques for handling these extensions, possible alternative implementations for distributed event detection, previous work on reasoning with ECA rules, and open questions relating to reasoning using the extended ECA rules.

Chapter 9, *"Pattern Detection in Extremely Resource-Constrained Devices"*, focuses on the challenges of detecting event patterns in distributed resource-constrained devices comprising a wireless network of sensors and actuators. Such pervasive computing environments pose particular problems due to the resource constraints of the devices and the lack of reliable communication and synchronisation capabilities in the network. The chapter discusses online data mining approaches to detecting anomalous or novel event patterns in the sensor data. The authors also present their own approach which combines concepts from data mining, machine learning and statistics and offers a variety of algorithms for both (i) detecting known patterns of interest in the sensor data, as specified by users, and (ii) detecting previously unknown event patterns by a phase of training on normal sensor data and then online detection of deviations from the learned patterns.

Chapter 10, *"Smart Patient Care"*, describes a prototype system for monitoring patients in emergency care units with the aim of predicting if they will have a cardiac arrest in the next 24 hours. The system merges real-time patient data with historical data, medical knowledge in the form of rules, and data mining models, in order to generate appropriate alarms customised according to the patient and also doctor's preferences. The system combines several Oracle products to achieve this functionality: Total Recall for managing the history of data changes, Continuous Query Notification for detecting changes in the data, Oracle Data Mining (ODM) for detecting complex patterns in the data, and Business Rules Manager for complex event processing.

Chapter 11, *"The principle of immanence in event-based distributed systems"*, focuses on collaborative Grid-based environments, and on the emergence of the principle of 'immanence' from the activities of collaborating partners in virtual organisations. It discusses how the AGORA Grid-based platform has been extended with agent-based mechanisms to foster the emergence of collaborative behaviours and to reflect information about the emergence of such behaviours back into the system architecture, with the aim of improving the self-organisation capabilities of the system and the collaborating community that it supports.

Chapter 12, *"Context-based event processing systems"*, discusses the role of context in event processing systems, identifying four main uses of context in such systems: temporal context, spatial context, segmentation-oriented context and state-oriented context. A survey is given of these different types of context. Events may be processed differently depending on their context. Event processing applications may use combinations of such context, and the chapter also discusses composing contexts. A survey is given of how context is supported in five commercial event processing systems.

The final chapter, *"Event Processing over Uncertain Data"*, identifies alternative approaches for capturing uncertainty in the detection of simple event occurrences and their attribute values, and in the inference of derived events. A taxonomy for event uncertainty is presented as is an analysis of the various causes of event uncertainty. Two models are proposed for representing uncertainty in simple events, based on probability theory and fuzzy set theory. Handling uncertainty in the inference of derived events is discussed with respect to a simple event language employing a Bayesian network inference model. Several open research questions are highlighted: identifying the most suitable approach, or approaches, for specific application requirements; specifying or automatically deriving appropriate inference rules and the probabilities associated with them; and achieving scalable implementations of the inference algorithms.

4 Concluding Remarks

This book aims to give readers an insight into the rapidly evolving field of reasoning in event-based distributed systems. Due to its fast, and sometimes uninhibited, growth this area faces the challenge of having to consolidate existing approaches and paradigms while at the same time integrating newly emerging application requirements, such as uncertainty, security and resource constraints. Nevertheless, the developments in this field have shown great potential for building effective and efficient distributed systems.

Logic-based approaches are being used for the specification and detection of complex events, giving a sound basis for reasoning about event properties and for scalable implementations of event detection mechanisms, possibly in combination with event stream processing middleware. Defining denotational semantics for event query languages provides an implementation-independent reference point for proving the correctness of implementations and developing optimisations. Data mining, machine learning and statistical approaches are being used to discover event patterns in high-volume event streams. Rule-based specifications of security policies make possible context-aware reasoning about information access and flow in distributed applications. Probabilistic approaches provide promising foundations for handling uncertainty in event detection and inference. Agent-based processing is being used to foster the emergence of online collaboration in social organisations through the interaction of intelligent agents that can adapt and evolve their behaviour over time.

The general approach of event-based distributed processing allows developers to create loosely-coupled and reconfigurable systems that offer scalability and adaptability, two of the most critical properties of modern information systems. Event processing can be monitored with respect to specific quality factors and this information be fed back into the system for dynamic quality improvement. Declarative languages for specifying resource management policies offer the potential for formal policy analysis and policy evolution. Rule-based mechanisms can be used to capture different users' needs and preferences in order to support personalisation of users' interaction with the system. Another important goal is the provision of versatile and powerful middleware components that support the rapid development of reliable distributed applications. Finally, we must state that event-based distributed processing and reasoning may still need some more time to mature, but we believe that it will be an exciting area to work in for the years to come.

References

1. Akyildiz, I.F., Weilian, S., Sankarasubramaniam, Y.: A survey on sensor networks. IEEE Communications Magazine 40(8), 102–114 (2002)
2. Bodoff, S., Armstrong, E., Ball, J., Carson, D.B.: The J2EE Tutorial. Addison-Wesley Longman, Boston (2004)
3. Chakravarthy, S., Adaikkalavan, R.: The Ubiquitous Nature of Event-Driven Approaches: A Retrospective View. In: Event Processing, Dagstul Seminar Proceedings 07191 (2007)

4. Chan, A.: Transactional publish/subscribe: the proactive multicast of database changes. ACM SIGMOD Record 27(2), 521 (1998)
5. Chen, J., DeWitt, D., Tian, F., Wang, Y.: NiagaraCQ: A scalable continuous query system for internet databases. In: ACM SIGMOD Conf. 2000, Dallas, TX, pp. 379–390 (2000)
6. Harrison, T., Levine, D., Schmidt, D.: The design and performance of a real-time CORBA event service. In: 12th ACM SIGPLAN Conf. on Object-Oriented Programming, Systems, Languages and Applications (OOPSLA), Atlanta, GA, pp. 184–200 (1997)
7. Hinze, A., Sachs, K., Buchmann, A.: Event-Based Applications and Enabling Technologies. In: 3rd ACM Int. Conf. on Distributed Event-Based (DEBS), Nashville, TN, pp. 1–15 (2009)
8. Levene, M., Poulovassilis, A. (eds.): Web Dynamics — Adapting to Change in Content, Size, Topology and Use. Springer, Berlin (2004)
9. Paton, N.W. (ed.): Active Rules in Database Systems. Springer, New York (1999)
10. Roussos, G.: Networked RFID: Systems, Software and Services. Springer, London (2008)
11. Widom, J., Ceri, S. (eds.): Active Database Systems: Triggers and Rules for Advanced Database Processing. Morgan Kaufmann, San Francisco (1994)