

Event Correlation: Language and Semantics

César Sánchez, Sriram Sankaranarayanan, Henny Sipma,
Ting Zhang, David Dill, and Zohar Manna *

Computer Science Department
Stanford University
{cesar,srirams,sipma,tingz,dill,zm}@CS.Stanford.EDU

Abstract. Event correlation is a service provided by middleware platforms that allows components in a publish/subscribe architecture to subscribe to patterns of events rather than individual events. Event correlation improves the scalability and performance of distributed systems, increases their analyzability, while reducing their complexity by moving functionality to the middleware. To ensure that event correlation is provided as a standard and reliable service, it must possess well-defined and unambiguous semantics.

In this paper we present a language and formal model for event correlation with operational semantics defined in terms of transducers. The language has been motivated by an avionics application and includes constructs for modes in addition to the more common constructs such as selection, accumulation and sequential composition. Prototype event processing engines for this language have been implemented in both C++ and Java and are now being integrated with third-party event channels.

1 Introduction

Publish-subscribe is an event-based model of distributed systems that decouples publishers from consumers. Consumers register their interests with the middleware by means of a subscription policy. Events published to the middleware are delivered only to those consumers that expressed an interest. The publish-subscribe paradigm is useful in practice for building large systems in which not all components are known at design time, or components may be dynamically added at runtime (such as in mobile systems). These systems may even extend world wide. Crowcroft et al [5] predict a world in which pervasive computing devices generate 10,000,000,000 events per second, with billions of subscribers from all over the planet. Clearly, extensive filtering and correlation are required at multiple levels and locations to manage these volumes.

Several middleware platforms provide event notification services. Examples include GRYPHON [1], ACE-TAO [19], SIENA [3], ELVIN [2]. An overview and comparison of such systems is in [15]. In some middleware platforms, such as the

* This research was supported in part by NSF grants CCR-01-21403, CCR-02-20134 and CCR-02-09237, by ARO grant DAAD19-01-1-0723, and by ARPA/AF contracts F33615-00-C-1693 and F33615-99-C-3014.

real-time event channel (RTEC) of ACE-TAO [19], consumers subscribe with the middleware with a list of event types or sources they wish to receive. Suppliers publish their events to the RTEC, which then distributes them to the subscribed consumers. This service is called event filtering.

Event correlation extends filtering with more complex subscriptions, including combinations of events and temporal patterns. Providing event correlation as a standard middleware service further enhances the performance of embedded applications. However, more importantly, it enables the transfer of functionality from application components to the middleware, which

- reduces software development and maintenance cost by decreasing the number of special-purpose components to be developed, as complexity is factored out of the component into the middleware;
- increases reliability, as this service can be verified and tested once as part of the platform and reused many times;
- increases the analyzability of the system, and accuracy of schedulability. Where event dependencies were before largely hidden inside application components, with event correlation provided, these dependencies are available explicitly to analysis tools in the form of event correlation expressions with well-defined semantics;
- increases flexibility in configuration: event correlation expressions can be changed on a shorter notice than components.

To enable event correlation to be provided as a standard, reliable middleware service, it must come with a well-defined, unambiguous semantics. In addition, it is desirable that the event-processing code be generated automatically from the (declarative) event correlation expressions. This ensures adherence to the semantics by construction.

In this paper, we present a language and a computational model for event correlation that provides an unambiguous semantics for event correlation expressions. The model is based on automata/transducers, a well-studied domain with a large body of analysis methods and tools. Another attractive property of transducers is that it is an operational model: they can be used directly in the RTEC to process events interpretatively, or used to automatically generate the code to process the events.

The development of this language was initiated to enable the use of event correlation in the Boeing Open Experimental Platform for the DARPA Information Exploitation Office Program Composition for Embedded Systems (PCES) program, based on Boeing's Bold Stroke avionics architecture [20]. Some of the constructs included in the language were directly motivated by Boeing product scenarios which illustrate important component level interactions in associated avionics applications.

The paper is organized as follows. In section 2 we give an intuitive overview of the features of a simple version of our event correlation expression language. In section 3 we present the new model of *correlation machines* and its extension *correlation modules* that are used to define the operational semantics of the constructs in the correlation language. This translation is described in section 4.