Chapter 1
Generic Architecture of Complex Event Processing Systems

Avigdor Gal
Israel Institute of Technology, Israel

Ethan Hadar
CA Labs, Israel

ABSTRACT
In recent years, there has been a growing need for the use of active systems, systems that are required to respond automatically to events. Examples include applications such as Business Activity Monitoring (BAM) and Business Process Management (BPM). Complex event processing is emerging as a discipline in its own right, with roots in existing research disciplines such as Databases and Software Engineering. This chapter aims at introducing a generic architecture of complex event processing systems that promotes modularity and flexibility. We start with a brief introduction of the primitive elements of complex event processing systems, namely events and rules. We discuss a grid approach to complex event processing systems. We detail the layers of the proposed architecture, as well as the architecture main components within the context of the major data flow in an event management system, namely: event collection, event purification, event storage, event inferencing, and event situation management. We discuss each of these elements in detail. Our tool of choice is the CA Agile Architecture (C3A) reference description approach (Hadar & Silberman, 2008). Throughout the chapter, we illustrate our discussion with two case studies. The first is that of service availability management, providing safeguards to critical business processes. The second involves disaster management, managing early warning systems.

INTRODUCTION
In recent years, there has been a growing need for the use of active systems. Active systems perform automatic actions that may be reactive, such as responding to provided stimuli. They may also perform automatic actions that are proactive, such as predicting possible phenomena. While earliest active systems were directed to databases (Dayal U. et al. 1988, Widom & Ceri, 1996), a current major need for such active functionality covers other areas such as Business Activity Monitoring
(BAM) (Balazinska et. al, 2008) and Business Process Management (BPM). Concurrent with the proliferation of such applications, the events to which active systems must respond have expanded from Information Technology (IT) systems and application-level events to business-level events. Active systems may encompass many levels of IT, from IT infrastructure events to business level objectives, and are used in a wide spectrum of sectors including financial and e-Trading, BPM, system management, client relationship management (CRM), workflow management, and military applications.

The main primitive in active systems is an event, a mechanism for delivering information. Events may be generated by some mechanism, external to the active system, and deliver information across distributed systems. Alternatively, events and their related data need to be inferred by the active system itself. Complex event processing (CEP) supports the inference of events based on event notification history and the data delivered by events. From an industrial point of view, CEP covers distributed a-synchronic message-based systems, such as enterprise applications that can immediately act upon business critical events.

Complex event processing is emerging as a discipline in its own right, with roots in existing research disciplines such as Artificial Intelligence, Business Process Management, Databases, Distributed Computing, Programming Languages, Semantics of Specification and Formal Verification methods, Sensor networks, Simulation, and Software Engineering. For example, in database research, events were suggested with the intention of introducing triggers in database management systems for monitoring application state and to automate applications by reducing or eliminating user intervention. This discipline still goes by various names such as Event Processing, Real-time Information Systems, Reactive Systems, Proactive Systems, and Active Technologies.

CEP requires intense computation efforts. Each complex event depends on its underlying events or previously calculated complex events. In addition, each of these complex events might be generated by domain specific business rules using this or that language. Therefore, CEP solutions must be scalable, flexible in configuration, easily managed, and efficient in terms of computation calculation. In this chapter we present a generic architecture for CEP systems adhering to the following principles:

1. A separation of concerns between different CEP instances that can interact in a form of a network of CEP instance components;
2. A simple modular and dynamic way to integrate CEP instances;
3. A means to reduce computation efforts within each of the CEP instances;
4. A centralized governance and compliance supportive arena, enabling CEP service virtualization and management capabilities;
5. A scalable and highly available solution, supporting real-time processing as well as analytics of historical events.

The rest of the chapter is organized as follows: We start with a presentation of two case studies to illustrate the architecture. Then, we present events and rules, and introduce the reader to the essentials of the C3A approach. Details of the architecture follow and we conclude with a discussion and future work.

**CASE STUDIES**

We begin with the introduction of two case studies to illustrate the architecture. The first is that of service availability management, providing safeguards to critical business processes. The second involves disaster management, and more specifically managing early warning systems.