Chapter 5
Modeling Context-Aware Distributed Event-Based Systems

Eduardo S. Barrenechea
University of Waterloo, Canada

Rolando Blanco
University of Waterloo, Canada

Paulo Alencar
University of Waterloo, Canada

ABSTRACT

Emerging applications are becoming increasingly dynamic, adaptive, and context-aware in areas such as just-in-time, location-based m-commerce; situational health monitoring; and dynamic social networking collaboration. Although numerous systems and implementation infrastructures have been proposed to deal with some features of such systems, there is a lack of higher-level modeling abstractions and associated component metamodels that combine dynamic features, adaptation mechanisms, and context-awareness. In this chapter the authors propose a metamodel for context-aware distributed event-based systems that supports novel forms of event-based context-aware interactions such as context event schemas; component interfaces that react to context, event, and context interactions at the interface level; and subscriptions based on both events and context. The approach also supports requirements for context-aware systems proposed in the literature. The applicability of the approach is illustrated by showing how existing context-aware systems can be modeled using the proposed metamodel.

INTRODUCTION

Recent emerging applications are becoming increasingly dynamic, adaptive, and context-aware in areas such as just-in-time location-based m-commerce, situational health monitoring, and dynamic social networking collaboration. Context-awareness provides systems with increased information about the situations in which its components and users are immersed. Context is highly dynamic, sometimes with drastic changes. For example, a change in location can have an effect on the status of the available services or behaviour of running applications. Context-aware

DOI: 10.4018/978-1-61520-655-1.ch005

systems are required to support this high level of dynamism.

Although numerous systems and implementation infrastructures have been proposed to deal with features of context-aware systems in isolation, there is a lack of component metamodels and higher-level modeling abstractions. Currently, context-awareness representation is typically addressed at the application level. The coupling between application and context-awareness related code introduces problems associated with software architecture and development, reusability, maintenance and program comprehension. By introducing context and context-awareness entities as first class citizens, separate from application level entities, it is possible to model context-awareness in an application-independent manner. This promotes reusability of concepts and features, as well as strengthens support for program comprehension, maintenance and architecture.

The features related with context-awareness require a highly dynamic infrastructure. Distributed event-based systems (DEBSs) provide such infrastructure through adaptive component interfaces and loosely coupled components communicating via event-based asynchronous interactions. The low coupling between components in a DEBS is due to the fact that components generating events are oblivious to components consuming the events and vice-versa. Consumers are interested in event types and not in the components generating events. This property of DEBSs allows context to be disseminated in a transparent fashion throughout the system. Context-aware components are notified of context related events generated from changes in context irrespectively of context sources. Context sources and context-aware components can enter and leave the system without the need to alter existing components.

In this chapter we propose a metamodel for context-aware distributed event-based systems. The metamodel supports the dynamism associated with context-awareness through event-based interactions and component interface changes. The metamodel consists of a structural view and a control view. The structural view is used to model the relationships between events and interfaces. The control view is used to model administrative components in a DEBS, as well as component grouping and access controls that can be imposed on events and interfaces. The model meets general requirements for context-aware systems proposed in the literature and later discussed in this report. The metamodel supports novel forms of event-based context-aware interactions, such as context event schemas, component interfaces that react to context, event and context interactions at the interface level, and subscription based on both events and context.

Distributed Event-Based Systems

A distributed event-based system (DEBS) is made up of independent functional components interacting with each other via events (Blanco et al., 2008). An *event* is a data representation of a happening in the system or the environment in which the system executes.

Events are generated by components called *publishers*. Components interested in the events that have been generated, are called *subscribers*. Other characteristics of DEBSs are:

- The kinds of events, components publish and subscribe to, can be introduced to, and removed from, the system at run time.
- A component publishes an event by explicitly invoking an *announce* or *publish* operation.
- Components must register their interest on the events they want to be notified about.
- When an event is published, the component announcing the event continues its execution without being blocked. The publisher component does not wait for subscriber components to be notified or for their reaction to the event.