Communication Paradigms for Mobile Computing

René Meier
rene.meier@cs.tcd.ie
Department of Computer Science, Trinity College Dublin, Ireland

The widespread deployment and use of wireless data communications causes the need for middleware to interconnect the components that comprise a mobile application. Middleware for mobile computing must deal with the increased complexity that comes with a dynamically changing population of application components and the resulting dynamic reconfiguration of the connections between these components. This paper presents an overview of two communication paradigms that are well suited as the basis for middleware for mobile computing, namely the event-based communication model and proximity-based group communication.

I. Introduction

The widespread deployment and use of wireless data communications in the mobile computing domain is generally recognized as being the next major advance in the information technology industry. Both mobility and wireless networking represent key enabling technologies underlying the vision of *ubiquitous computing* [11], where interconnected computers will be embedded in a wide range of appliances ranging in size from door locks to vehicle controllers performing tasks on behalf of their human users, such as automatically opening doors and routing vehicles to their intended destinations.

Emerging mobile and ubiquitous computing applications in various domains including indoor and outdoor smart environments, augmented reality, and traffic management typically comprise of a large number of interconnected components distributed over a vast geographical area. Middleware for application component integration must deal with the increased complexity that comes with such scale and the geographical dispersion of the components. Most significantly, middleware for mobile computing must accommodate the dynamically changing population of components in a certain geographical area, which results in connections between components being established very dynamically during their lifetime [9]. Thus, such middleware must support communication paradigms that allow unanticipated interactions between application components enabling them to dynamically establish connections to other components within their current vicinity. In this paper, we summarize the results of our initial exploration into communication paradigms that are suitable for mobile computing as presented at the Advanced Topic Workshop on Middleware for Mobile Computing.

II. Event-Based Communication

The *event-based communication model* represents an emerging paradigm for middleware that asynchronously interconnects [1] the components that comprise an application in a potentially distributed and heterogeneous environment, and has recently become widely used in application areas such as large-scale Internet services and mobile programming environments. The event-based communication model supports a one-to-many or many-to-many communication pattern that allows one or more application components to react to a change in the state of another application component. *Event notifications*, or simply events, are the messages that contain the data representing the change to the state of the sending component. They are propagated from the sending components, called the producers, to the receiving components, called the consumers. Events typically have a name and may have a set of typed parameters whose specific values describe the specific change to the producer's state. A particular consumer may only be interested in a subset of the potentially large number of events propagated in a system. *Event filters* provide a means for a consumer to subscribe to the (ideally) exact set of events that it is interested in receiving. Before events are propagated, they are matched against the filters and are only delivered to consumers that are interested in them, i.e., for which the matching produced a positive result.

Event-based communication is well suited to addressing the requirements of the mobile computing domain [4]. It avoids centralized control as well as long-lasting and hence potentially expensive connec-

tions and requires a less tightly coupled communication relationship between application components compared to the traditional client/server communication model. The notion of dynamically inaugurating communication relationships among components without relying on centralized control is central to addressing the needs of a scalable system, representing the ability to accommodate growth in a potentially large-scale distributed environment. Furthermore, filtering capabilities typically associated with event-based communication improve scalability by decreasing network traffic [5].

Scalable Timed Events And Mobility (STEAM) [8] is an event-based middleware service that has been designed for mobile applications such as traffic management. STEAM targets application scenarios that include a large number of application components, called entities, representing real world objects that communicate using wireless technology and the ad hoc network model. Many of these entities may represent mobile objects including cars and ambulances; other entities may represent objects with a fixed location, such as traffic signals and lights. These entities interact using event-based communication in order to exchange information on the current traffic situation. For example, a traffic signal may propagate a change to the speed limit due to road conditions to approaching cars. Another example may involve an ambulance disseminating its location to the cars in its vicinity for them to yield the right of way.

Notably, STEAM is motivated by the hypothesis that in this type of application scenario entities are most likely to interact once they are in close proximity. This means that the closer consumers of events are located to a producer of events the more likely they are to be interested in the events propagated by that producer. Significantly, this implies that events are valid within a certain geographical area surrounding a producer. An example scenario demonstrating such behavior would be a traffic light disseminating its status and cars being interested in receiving these events only if they are located within a certain range of the light. This approach to propagating events within a certain area surrounding producers limits forwarding of event messages, and therefore reduces the usage of communication and computation resources, which are typically scarce in mobile environments. Furthermore, it reduces the complexity of handling the geographical dispersion of the entities and the scale of the system. A dynamically changing population of entities in a certain geographical area of the system is relevant to that area, requiring the affected area of the system to reconfigure while reconfiguration of other areas will be limited.

III. Proximity-Based Group Communication

Classical *group communication* [3] provides a one-to-many or many-to-many communication pattern typically based on a reliable multicast protocol that allows a member of a group to send messages to all members of that group. This communication pattern can be used by producers to propagate messages to a group of consumers exploiting the delivery semantics associated with the group and therefore has been recognized as a natural means to support event-based communication models [2].

An extension to classical process groups called proximity groups has been identified as a useful communication paradigm for mobile applications [6]. Proximity groups allow potentially mobile application components to join a proximity group and subsequently interact with its members once they are within the same geographical area. Related research in this area has focussed on routing protocols for group communication based on geocast [7] as well as on communication groups in which membership is solely based on the location of application components [10]. Significantly, this notion of proximity groups defines membership by both functional and geographical aspects. The functional aspect, i.e., the name of the group, represents the common interest of group members based on the information that is propagated among them. The geographical aspect, i.e., the geographical area, outlines the scope within which the information is valid. In order to apply for proximity group membership, an application component must firstly be interested in the group and secondly be located in the geographical area that corresponds to the group. In contrast, classical group communication defines groups solely by their functional aspect. Additionally, a proximity group can be distinguished as either absolute or relative. The geographical area associated with an absolute proximity group is geographically fixed, whereas the geographical area associated with a relative proximity group is relative to a moving point in space, most likely one of the proximity group's mobile members.

Proximity groups are exploited by STEAM as the underlying means for application components to interact. STEAM utilizes the functional aspect of proximity groups to define the common interest of producers and consumers based on the type of information that

is propagated among them and the geographical aspect to outline the scope within which the information is valid, i.e., the area within which the corresponding events are propagated. In addition, STEAM exploits the message delivery semantics associated with proximity groups to provide end-to-end guarantees when delivering events.

References

[1] BACON, J., MOODY, K., BATES, J., HAYTON, R., MA, C., MCNEIL, A., SEIDEL, O., AND SPITERI, M. Generic support for distributed applications. *IEEE Computer 33*, 3 (2000), 68–76.

[2] BANAVAR, G., CHANDRA, T., STROM, R., AND STURMAN, D. A case for message oriented middleware. In *Proceedings of the 13th International Symposium on DIStributed Computing (DISC'99)* (Bratislava, Slovak Republic, 1999).

[3] CRISTIAN, F. Synchronous and asynchronous group communication. *Communications of the ACM 39*, 4 (1996).

[4] CUGOLA, G., NITTO, E. D., AND FUGGETTA, A. The JEDI event-based infrastructure and its application to the development of the OPSS WFMS. *IEEE Transaction of Software Engineering (TSE) 27*, 9 (2001), 827–850.

[5] HAAHR, M., MEIER, R., NIXON, P., CAHILL, V., AND JUL, E. Filtering and scalability in the eco distributed event model. In *Proceedings of the 5th Int. Symp. on Software Engineering for Parallel and Distributed Systems (ICSE/PDSE2000)*. IEEE Computer Society, Limerick, Ireland, 2000, pp. 83–95.

[6] KILLIJIAN, M. O., CUNNINGHAM, R., MEIER, R., MAZARE, L., AND CAHILL, V. Towards group communication for mobile participants. In *Proceedings of Principles of Mobile Computing (POMC)*. Newport, Rhode Island, USA, 2001, pp. 75–82.

[7] KO, Y.-B., AND VAIDYA, N. H. GeoTORA: A protocol for geocasting in mobile ad hoc networks. In *Proceedings of the 8th International Conference on Network Protocols (ICNP)*. Osaka, Japan, 2000.

[8] MEIER, R., AND CAHILL, V. STEAM: Event-based middleware for wireless ad hoc networks. In *Proceedings of the International Workshop on Distributed Event-Based Systems (ICDCS/DEBS'02)*. Vienna, Austria, 2002.

[9] NIXON, P., AND CAHILL, V. Mobile computing: Technologies for a disconnected society. *IEEE Internet Computing 2*, 1 (1998), 19–21.

[10] ROMAN, G.-C., HUANG, Q., AND HAZEMI, A. Consistent group membership in ad hoc networks. In *Proceedings of the 23rd International Conference on Software Engineering (ICSE)*. Toronto, Canada, 2001.

[11] WEISER, M. The computer for the 21st century. *Scientific American 265*, 3 (September 1991), 94–104.