Abstract

In ubiquitous computing research it is generally accepted that software has to be loosely-coupled and event-driven. Applications have to be decoupled in space, time and arity in order to cope with dynamically reconfiguring wireless networks and intermittent network connections. They have to be event-driven in order to react to fluctuating wireless connections, users, other computers, sensors, etc. This has led to a number of programming models tailored towards such distributed event-driven applications. The one on which this work is based is the ambient-oriented programming paradigm [DVCM05], which is a programming language paradigm specifically tailored towards applications running on mobile devices, such as smartphones.

RFID is an emerging wireless technology that allows to enrich everyday objects with a digital representation. Current RFID-enabled applications merely use RFID tags as a barcodes, associating physical objects with a digital representation in a centralized backend database. However, RFID technology is increasingly integrated in mobile devices, giving rise to mobile RFID-enabled applications. We begin this dissertation by extending ambient-oriented programming to be able to conceive mobile RFID-enabled applications as ambient-oriented programs running on mobile devices where remote software objects can designate tagged real-world things.

Like any event-driven architecture, ambient-oriented programming requires the programmer to coordinate an application by means of callbacks, which causes the well-documented problem of inversion of control [HO06]. This problem becomes much more prominent in mobile RFID-enabled applications because of the massive number of events that such applications need to react to. This dissertation proposes dataflow programming in order to tackle this problem. Dataflow programming allows reacting to events without inversion of control [MRO10]. Unfortunately, current incarnations of dataflow programming do not offer the aforementioned decoupling in space, time and arity. To achieve this decoupling, we integrate dataflow programming with ambient-oriented programming into ambient-oriented dataflow programming.

We analyze ambient-oriented dataflow programming on two levels, namely the node level and the network level. With node-centric ambient-oriented dataflow programming, nodes specify their distributed dataflow dependencies declaratively by subscribing to and publishing events, and let the language runtime take care of coordinating dependent event-driven code instead of relying on callbacks. However, since dependent code is evaluated implicitly in response to remotely signaled events, the global control flow of a large distributed applications becomes implicit and hard to predict. This leads us to design a network-centric visual ambient-oriented dataflow language in which an explicit network-level data flow is used for coordinating an application. In order to cope with the dynamic nature of mobile RFID-enabled applications, it allows to visually specify the arity and binding semantics of distributed dataflow dependencies.