

Jorge Vallejos

ABSTRACT

The pervasive computing field envisions users surrounded by computers embedded in everyday devices. Such devices form dynamic networks which change topology as users move about. Software services are expected to maximise available computational capacities by seamlessly coordinating with each other. The services should also react to dynamic changes in the environment and adapt their behaviour accordingly. An increasingly popular solution to cope with such requirements is to represent environmental changes as events. Thus, pervasive computing software is often modelled as event-driven systems. However, such support typically comes at a price in the evolvability of programs. The more events a service has to be aware of, the more cumbersome its control flow becomes. The lack of adequate language abstractions to handle such events results in programs that are difficult to maintain and extend.

In this work, we study the effects of event-driven distribution on the modularity of programs. We focus on the modularity of two concerns: the capacity of services to adapt to environmental changes —a property known as context-dependent behaviour— and their capacity to coordinate with each other —a property known as group behaviour. We use the ambient-oriented programming (AmOP) paradigm as the basis of our research. This paradigm proposes an event-driven programming model which has been designed specifically for pervasive computing. We identify a list of requirements for a unified model for event-driven distribution, context dependency and group behaviour. In the study of the state of the art we demonstrate that no single approach fulfils these requirements so far. This observation has led us to the definition of a proof-of-concept programming language model, called Lambic.

Lambic is an extension of the generic function-based object system of the Common Lisp programming language. Our model extends the multiple dispatch semantics of generic functions to allow for modularity in pervasive computing. For event-driven distribution, Lambic integrates the properties of the AmOP paradigm, in what we call futurised generic functions. In addition, we provide a novel mechanism to allow flexible selection and composition of behaviour, called predicated generic functions. This mechanism provides language abstractions to influence method dispatch based on the program's context. For group behaviour, we propose a third extension called group generic functions. The main idea of this feature is to encapsulate the coordination of group interactions in the definition of services. Finally, a common underlying execution process ensures that these three features can be effectively used in combination with one another. We validate our work by showing in a number of case studies how Lambic facilitates the natural integration and dynamic adaptation of pervasive computing services.

SAMENVATTING

In de visie van pervasive computing zijn computers ingebed in alledaagse apparaten. Zulke apparaten vormen onderling dynamische communicatienetwerken waarvan de topologie verandert naarmate de gebruikers zich verplaatsen. Van software services wordt verwacht dat ze maximaal gebruik maken van de beschikbare apparaten en dat de vereiste coördinatie transparant verloopt. Deze services moeten ook kunnen reageren op dynamische veranderingen in de omgeving zodat ze hun gedrag automatisch kunnen aanpassen. Een populaire aanpak om aan deze vereisten te voldoen bestaat erin om veranderingen in de omgeving voor te stellen als events, zodat pervasive computing applicaties gemodelleerd worden als event-gedreven systemen. Deze strategie heeft echter wel een impact op de onderhoudbaarheid van pervasive computing programma's. Hoe meer verschillende events een applicatie moet ondersteunen, hoe complexer het wordt om de control flow van het programma te beheren. Door een gebrek aan geschikte taalconstructies om events te manipuleren, is het in het algemeen moeilijk om zulke programma's te onderhouden en uit te breiden.

In deze verhandeling bestuderen we de impact van event-gedreven distributie op de modulaire structuur van programma's. We bestuderen twee bekommernissen in detail: Enerzijds bestuderen we de mogelijkheid van programma's om zich aan te passen aan veranderingen in de omgeving—een eigenschap die bekend staat als contextafhankelijk gedrag—en anderzijds bestuderen we de capaciteit van programma's om met elkaar te coördineren—een eigenschap die bekend staat als groepsafhankelijk gedrag. We baseren ons onderzoek op het ambient-georiënteerde programmeerparadigma (AmOP). Dit paradigma definieert een event-gedreven programmeermodel dat specifiek ontworpen is voor het pervasive computing domein. We identificeren een aantal vereisten voor het unificeren van event-gestuurde distributie, contextafhankelijk gedrag, en groepsafhankelijk gedrag. In ons overzicht van de literatuur tonen we aan dat geen van de bestaande programmeermodellen aan al deze vereisten voldoet. Dit leidde ons ertoe een experimentele programmeertaal te ontwikkelen voor pervasive computing, genaamd Lambic.

Lambic is een uitbreiding van het op generische functies gebaseerde objectmodel van de programmeertaal Common Lisp. Ons model breidt de multiple dispatch semantiek van generische functies uit om een modulaire programmastructuur mogelijk te maken voor pervasive computing applicaties. Ter ondersteuning van event-gestuurde distributie, vervult Lambic de eigenschappen van het AmOP paradigma via een mechanisme dat we futurised generic functions noemen. Daarbovenop bieden we een nieuw mechanisme aan om de flexibele selectie en samenstelling van gedrag toe te laten. Dit mechanisme noemen we predicated

generic functions. Het laat ons toe om taalconstructies te definiëren zodat method dispatch de dynamische context van het programma in acht neemt. Ter ondersteuning van groepsafhankelijk gedrag, stellen we een derde uitbreiding voor, met name group generic functions. Het idee achter dit mechanisme is om de coördinatie van interacties tussen groepen in te kapselen in de definitie van services. Tot slot hebben we een gemeenschappelijk uitvoermodel gedefinieerd zodat de drie mechanismen effectief in combinatie met elkaar gebruikt kunnen worden. We valideren ons onderzoek door het bespreken van een aantal case studies die aantonen hoe Lambic het mogelijk maakt om vlot integreerbare en dynamisch aanpasbare pervasive computing services te ontwikkelen.