IPASJ International Journal of Electrical Engineering (IIJEE)
Web Site: http://www.ipasj.org/IIJEE/IIJEE.htm
A Publisher for Research Motivation........
Email: editoriijee@ipasj.org
Volume 5, Issue 2, February 2017
ISSN 2321-600X

RAISED AREA SELF-GOVERNING, HIGHER-ORDER, STATICALLY CHECKED MOBILE APPLICATIONS

Miss. Vinita Suman

Indian Institute of Technology, Kharagpur

ABSTRACT

There is increasing interest in establishing a presence in the mobile application market, with platforms including Apple iPhone, Google Android and Microsoft Windows Mobile. Because of the variations in platform languages, frameworks, and device hardware development of an application for additional than one platform will be a troublesome task. In this paper we address this downside by the creation of a mobile Domain Specific Language (DSL). Domain analysis is carried out using 2 case studies, leading to the identification of basic requirements for the language. The language is defined as associate extension of a λ- calculus in terms of associate operation linguistics and a kind system. The language is a contribution to the understanding of mobile applications since it precisely defines the essential properties offered by a spread of mobile application technologies, and can type the idea for one language that may target multiple platforms.

I. INTRODUCTION

TODAY, the penetration of modern sensible phones is immensely increasing with over 172 million sensible phones shipped worldwide in 2009 [1], and with the emergence and successes of sources for customers to install third party applications opens a brand new marketplace for developers to achieve consumers. However, developing an application for multiple mobile platforms will incur totally different obstacles as well as variations in development tools out there, different languages, platform constraints and availability of software package libraries. Difficulties in producing software package for additional than one platform has been evident for several years outside of the mobile realm. For decades, software movableness has been a concern throughout development, mainly due to terribly giant spectrum of various electronic equipment Instruction Set Architectures (ISA) and by the massive style of operative Systems in use. Recently this has become less of a problem, largely due to factors as well as the decrease in electronic equipment ISAs, the dominance of a limited variety of operative systems, and to commonly used languages as well as Java. Since the mobile market is relatively immature, there are giant variations in implementation languages and development environments used for totally different applications and technology platforms.

A. Outline of Paper

The outline of this paper is as follows: section II describes the background to mobile application development; section III outlines current approaches to multiplatform strategies and defines our contribution; section IV describes 2 case studies that we have a tendency to have enforced for an organization and that ends up in the identification of the domain options for our language; section V defines the syntax of the language and provides an easy example application; section VI defines however the language executes; finally section VII analyses the language and describes our current and future implementation methods.

II. BACKGROUND AND RELATED WORK

A Software Development Kit (SDK) provides the atmosphere for developing applications through the use of libraries, emulators, and debuggers and is the basis for the event of the bulk of current mobile applications. However associate unfortunate disadvantage of SDKs is that they ar platform dependant: Apple provides associate Xcode SDK (http://developer.apple.com/devcenter/ios) that includes an interface builder, associate iPhone machine and development environment; the automaton SDK comes as an eclipse plug-in, and includes a software emulator; the Windows Phone offers a specialised version of the Microsoft Visual Environment; Blackberry and Symbian additionally provide totally different SDK platforms. This variance in terms of specific platforms greatly increases the prices of developing mobile applications to be accessible in an exceedingly style of devices. In order to cut back these costs, various frameworks have been projected so as to provide cross-platform applications.

A. Frameworks

The DIMAG Framework [2] was developed for automatic multiple mobile platform application generation. This was accomplished by creating a declarative definition language that is comprised of three distinct parts; first off a language DIMAG-root, which provides references to the definitions for progress and user interface within the application; second the language State Chart protrusible nomenclature (SCXML) defines the progress by the definitions of states, state transitions, and condition based actions; and finally DIMAG-UI language supported MyMobileWeb's IDEAL language victimisation CSS to regulate the interface.The main shortcomings of this method is that it depends on server-side code generation and transfer. The other distinction with our work is that applications developed during this framework ar understood employing a virtual machine. Other frameworks embrace XMLVM [3], [4] developed at San Francisco State University and created to support byte-code cross-compilation and avoid source-code translation through the employment of a tool chain. This tool chain currently interprets Java category files and .Net viable to XML documents, which then will be output to Java computer memory unit code/.NET CIL or to JavaScript and Objective-C. This tool chain was firstly used to cross compile Java applications to Ajax applications [5], because of the dearth of IDE support associated issue in making an Ajax application. Further work to embrace automaton to iPhone application cross-compilation [6] was completed. API mapping between the two platforms was carried by the creation of a compatibility library. More recently, since the arrival of HTML5 and WebKit, a number of open supply and business cross-platform frameworks are projected like the Appcelaterator (http://developer.appcelerator.com), PhoneGap (http://www.phonegap.com) and Rhomobile (http://rhomobile.com). Frameworks, which use either JavaScript or Ruby, and therefore the ensuing applications ar run in an exceedingly browser. Furthermore these applications will run offline and access the device's full capabilities; such as a GPS or camera; providing an equivalent look and feel as a native application. Although these frameworks greatly change the task of implementing the mobile application, the developer is still required to figure with general net application languages that lack specificity and potency in terms of mobile applications. Furthermore, these applications suffer limited visibility on the market due to the absence of associate "official" marketing. There are many totally different software package platforms for mobile applications. Since commercial developers sometimes would like to develop associate application {that will|which will|that may} beat up all platforms there ar variety of proposals for single technologies that can target multiple platforms. A recent proposal for a subscriber line for mobile applications [7] uses XText and Eclipse to implement a DSL that uses code generation techniques to target mobile platforms. This DSL uses mounted user interface structures such as section whereas our language leaves the gathering of external widgets as a parameter of any use of the system. It is also not clear whether or not the subscriber line includes a static kind system and its linguistics isn't outlined severally of a translation to a target platform.

B. Domain Specific Languages and Modelling

Domain-specific languages (DSLs) have provided the support for software development method by raising the level of abstraction and introducing specialised viewpoints of a definite downside house. The benefit of subscriber line to application development has been represented in [10], [11], [12], [13]. More recent DSLs in alternative areas embrace [14] that concentrates on the abstraction of net applications to lower the general complexness of the applying and boilerplate code. Further work on this subscriber line diode to the creation of Platform freelance Language (PIL) [15]. PIL was developed as an intermediate language, to provide a scalable technique for developing for multiple platforms. A drawback of this technique is presently it lacks support for mobile platform development. Other efforts for creating mobile application development easier embrace Google straightforward (http://code.google.com/p/simple/), a BASIC dialect for making automaton applications, and more recently the Google App discoverer (http://appinventor.googlelabs.com/about/), which is based mostly on Openblocks [16] and Kawa (http://www.gnu.org/software/kawa/). Particularly Google App discoverer has immensely abstracted app development, but solely supports development of automaton applications.

More recently, Brenhs has proposed MDSD, a DSL for iPhone. The language is more specific to information central applications. Following from that work, they have started the Applause project for developing subscriber line for iPhone, iPad and Android (http://code.google.com/p/applause/), but this is still not absolutely developed. The SERG group outlined a language named Mobl (http://www.mobl-lang.org/) with a declarative DSL for each the user interface and for outlining the information. Although mobl is comparable to our language there ar substantial variations since we've got aimed to capture the essential options of mobile applications through the employment of: technology independence; static writing for all features; higher-order functions; formally outlined operational semantics; gizmo libraries. Our aim is for the language presented in this article to be a proper foundation for every of this implementations of mobile applications DSLs.

III. MULTI-PLATFORM DEVELOPMENT AND CONTRIBUTION

Because of the complexities in multiplatform development, in this section we introduce 3 strategies that would be accustomed facilitate application development for multiple platforms. Our proposal is to select associate approach supported Domain Specific Languages.

A. Approaches to Development

Frameworks: The use of frameworks will be seen as a technique of software abstraction victimisation common code, which will be overridden and extended by a user. Within mobile development, frameworks have been developed to assist with specific tasks including media playback, access to sensors and graphic and UI manipulation. For example, the system described in [6] has been designed to facilitate build code bindings between the various platform specific frameworks. This method concentrates on finding all procedure issues, which will increase complexness in application development, further turning into a hindrance to the developer.

Web Applications: A mobile net application primarily is a regular web application designed to suit the common screen sizes of most mobile devices, bringing varied edges to the developer. Some applications that require high amounts of process will greatly get pleasure from permitting the process to be handled within the cloud whereas the device just must method the UI. In addition the employment of standards like HTML and CSS might make sure styles of applications easier to develop. Originally this approach was troublesome as a result of of reliance on network property for the applying, which in some things might either not be out there or not desired. Web applications additionally couldn't store native information to the net browser, until the development of HTML5 [18]. In May 2007, Google released a plug-in for the Firefox net browser, Google Gears (http://gears.google.com/). This plug-in supports caching of web applications to enable offline use, and also the ability for an online application to store information in an exceedingly native information. This idea has been integrated into the event of HTML5, a step in the right direction but there ar still remaining problems. Firstly, web applications in general will have shortcomings within the quantity of wealthy UI widgets, with animation for certain gizmo interaction being increasing troublesome to implement in an exceedingly mobile net application. Other issues ar limitations of the web-browser on the mobile devices, possibly leading to inconsistencies in application practicality between {different|totally totally different|completely different} platforms due to lack of API for victimisation different device elements (e.g. accelerometers, vibration motors, GPS etc). Because of these limitations and current unsoluble dependencies, the creation of a Domain Specific Language was chosen as our solution.

Domain Specific Languages: A Domain Specific Language (DSL) [19], [20] is primarily designed to be used in a definite application domain (e.g. mobile, telecoms, finance), abstracting away from the software implementation creating implementation easier. The abstraction is designed to help the developer and is to be contrasted with General Purpose Languages (GPLs) whose features aren't designed with any explicit domain in mind. DSLs have existed for many years. Languages that were created for particular domains embrace algebraic language [21] used to enable direct mathematical formula, Structured Query Language (SQL) [22] for information access and manipulation, and Algol [23] for algorithm specification. In recent times, the use of DSLs are proposed and utilized in totally different domains as well as the assembly of wealthy net applications [14], mashups of web applications and services [24], and system integration [25]. Because of the complexities in mobile development, we believe there is area for abstraction within the development for mobile devices.

IV. DOMAIN ANALYSIS

A DSL is outlined by acting a domain analysis [20] on a target family of applications so as to spot the common characteristic options. The domain analysis leads to the look of a technology that conveniently supports these options. Our aim is to define a language that will be accustomed represent mobile applications and thus our domain analysis starts with the development and analysis of 2 phone applications developed as a part of University business and enterprise activities. This section describes the case studies and then outlines the characteristic features that were known.

A. Case Studies Two iPhone application case studies were created for a native tiny to Medium sized Enterprise (SME). These applications are represented in the following 2 sub-sections.

Tour de France (TDF2009): This application was created to offer access to data from the 2009 series Tour Diamond State France cycle race. Firstly the application needed a technique of transferring associated receiving information from an external server for 2 totally different reasons. Firstly for the stage results, and secondly for the general information as well as data regarding the Teams/Riders and every one the Stages concerned in this year, this helped us come through a terribly tiny installation size. The data communications were done via XML files parsed victimisation the iPhone SAX-XML programme, one created with the static data, and one generated every day with this results. Inside the stages section, fly-through videos to help illustrate the course and parcel of land, large high resolution gesture controlled photos were incorporated. Key features of the TDF2009 ar shown in Figure one wherever a main screen provides access to the results of current stages and to fly-through videos. The figure shows that the applying is driven by events caused by the user touching the screen which the application consists of various displays (or states) consisting of a combination of event processors (buttons) and straightforward information (images, text).

REFERENCES

[1]. H. J. De La Vergne, C. Milanesi, A. Zimmermann, R. Cozza, T. H. Nguyen, A. Gupta, and C. Lu, “Competitive landscape: Mobile Devices, Worldwide, 4Q09 and 2009,” Gartner., Stamford, CT, Tech. Rep., Feb. 2010.
[2]. P. Miravet, I. Marín, F. Ortín, and A. Rionda, “Dimag: A Framework for Automatic Generation of Mobile Applications for Multiple Platforms,” in Proc. of the 6th International Conference on Mobile Technology, Application &amp; Systems, 2009, pp. 1–8.
[3]. A. Puder, “An XML-based Cross-Language Framework,” in Proc. On the Move to Meaningful web Systems 2005: OTM 2005 Workshops, LNCS vol. 3762, Springer, 2005, pp. 20–21.
[4]. A. Puder, “A Code Migration Framework for Ajax Applications,” in Proc. Distributed Applications and Interoperable Systems, half dozenth IFIP WG 6.1 International Conference, LNCS vol. 4025, Springer, 2006, pp. 138– 151.
[5]. A. Puder, “A Cross-Language Framework for Developing Ajax Applications,” in PPPJ ’07: Proc. of the 5th International conference on Principles and follow of Programming in Java, 2007, pp. 105–112.
[6]. A. Puder and I. Yoon, “Smartphone Cross-Compilation Framework for Multiplayer Online Games,” in Proc. of the International Conference on Mobile, Hybrid, and On-line Learning, 2010, pp. 87–92.
[7]. H. Behrens, “MDSD for the iPhone: Developing a Domain-Specific Language and IDE Tooling to Produce globe Applications for Mobile Devices,” in Splash ’10, Proc. of the ACM International Conference Companion on Object Oriented Programming Systems Languages and Applications Companion, 2010, pp. 123–128.
[8]. F. Balagtas-Fernandez and H. Hussmann, “Evaluation of User-Interfaces for Mobile Application Development Environments,” in Proc. of the 13th International Conference on Human-Computer Interaction. Part I: New Trends, 2009, pp. 204–213.
[9]. E. Cooper, S. Lindley, P. Wadler, and J. Yallop, “Links: Web programming while not Tiers,” in Proc. of the 5th International conference on Formal strategies for elements and Objects (FMCO), 2006, pp. 266– 296.
[10]. J. R.M. Herndon and V. Berzins, “The Realizable Benefits of a Language Prototyping Language,” IEEE Transactions on Software Engineering, vol. 14, no. 6, pp. 803–809, 1988.