ISSN: 2321-7782 (Online)
Volume 2, Issue 12, December 2014
International Journal of Advance Research in Computer Science and Management Studies
Research Article / Survey Paper / Case Study
Available online at: www.ijarcsms.com

Cloud: Native Platform for Mobile Applications

Bhakti Bhatti[1]
computer science
SCSCOE, Dhangwadi, Bhor
Pune, India

Vidya Chavan[2]
computer science
SCSCOE, Dhangwadi, Bhor
Pune, India

Shital Dhanke[3]
computer science
SCSCOE, Dhangwadi, Bhor
Pune, India

Rupali Tandel[4]
computer science
SCSCOE, Dhangwadi, Bhor
Pune, India

Bhagwan D. Thorat[5]
Asst. Professor
Computer Department
SCSCOE, Dhangwadi, Bhor
Pune, India

Abstract: ***To survive in computing market, companies or open source organization often have to release version of their project in different language. Manually porting application written for source platform to target platform. To reduced time, effort an error, tools can be developed for automatic mapping o project from one language to another language. However, this task is difficult because developer must know the knowledge of mapping between methods in source and target APIs, referred to as API mapping relations .In this paper, we develop a novel approach to problem of transferring between the APIs of source and target platform. Our approach is solution for a mapping between source APIs to target APIs where source and target APIs each have independently-develop application that implement similar functionality .The output of our approach shows that our tools is unique for mapping relationship APIs between source and target platform.***

I. INTRODUCTION

As world is becoming business world so, to survive in market environment many companies and organizations launches their projects with different languages. The current condition of market for mobiles is the applications which are launches everyday are available on variety of platforms. Every software engineer often wishes to make his app available on various Smartphone's, tablets and desktop. There is great benefit by making available mobile applications on various platforms. So that they can maximize the customer base.

For successful migration of applications we need to port the apps from one platform to another platform. But the porting of applications it difficult problem. To recover this problem the solution is parented in this paper, we present a solution to the heterogeneity problem of the mobile application market that does not require manual porting of applications nor shifting the development into cross-platform frameworks. Our solution, called Cloud Twin, makes it possible to execute mobile applications written for one platform natively on another platform.. To overcome the problem of heterogeneity the designers have created framework for cross platform mobile development, such as Phone Gap [4]. Mapping projects from one language to another language manually is monotonous and error-prone task. We present an approach to automate the creation of mapping databases for any given source/target API pair. A source platform application S and a target platform application T, possibly developed independently by different sets of programmers, constitute a similar application pair if they implement similar high-level functionality. Our approach works by recording traces of S and T executing similar tasks, structurally analyzing these traces, and extracting likely mappings using probabilistic inference. The output of our approach is a ranked list of mappings inferred for each source API method [3].

© 2014, IJARCSMS All Rights Reserved

II. DEFINATIONS

API: An Application Programming Interface [2][5] is set of classes and function, methods provided by frameworks or libraries.

API Library: An API library is a framework or library that provides reusable API classes and methods [5].

UI: The user interface (UI) is everything designed into an information device with which a human begin may interact

Mapping Relation: For entities E1 (like API classes and functions) in a language L1 and entities E2 in another language L2, mapping relation is triple (E1,E2,b) where mapping between E1 and E2 maintains the b i.e. behavior. The b is specific to the type of the entities [5].

III. APPROACH

Cloud Twin first automatically translates the initial screens of the source. Then it continuously captures, transmits, and replays user actions and UI updates, until the user switches to another application. In the following discussion, we detail each major component of Cloud Twin in sequence.

Step1:[3] In the first step we need to gather a database of applications i.e. collection of application pair. For each source application we need a target application that implement the same high level functionality.

Step2: At the source application (android), Cloud Twin intercepts the execution point where the UI is created but not yet displayed to the user. Then cloud twin examines the runtime UI tree for detection of the constituent interface components and the listeners attached to them. The interception is accomplished by means of Aspect-Oriented Programming (AOP) [1] and the UI tree is walked by means of reflection.

Step3: Cloud Twin represents the running source application as a collection XML structures. The traces in each API were obtained by exercising similar functionality in the source and target applications, these traces must contain some API calls that can be mapped to each other. This is the key intuition underlying our approach.. The output of this step is a database of functionally equivalent trace pairs ($Trace_S$, $Trace_T$) across all of the application pairs collected in Step 1.

Step4: Cloud Twin utilizes the XML intermediate form created by the source application to subsequently generate a target language application. In the reference implementation of Cloud Twin, Windows Phone applications are created by generating XAML and XAML.CS files from the intermediate XML files.

Step5: The final major component of Cloud Twin is executing the target application by communicating with the source's emulator. The reference implementation of cloud twin natively emulates the behavior of android apps on the windows phone. Cloud twin transmit via web socket, the UI action perform on windows phone to the cloud server, which mimics the received action on the android emulator. The UI updates on the emulator are efficiently captured by means of AOP and sent back to be replayed on the Windows Phone.

IV. MATHEMATICAL INDUCTION

CE ()-cloud emulator

E= E set of output events (e_1,e_2,e_3……..e_n); triggered on source UI

OE=OE set of output event (oe_1,oe_2,oe_3…oe_n) executed on target platform

CE= it is a cloud emulator, composed from XML representation CE takes the event generated on source UI as input and convert it to the output events to be executed on target platform.

© 2014, IJARCSMS All Rights Reserved

Mathematically we can represent emulator functionality as follows:

CE (E_n) =OE_n

Where

CE- cloud emulator

E= {e_1,e_2,e_3…e_n}

OEn= {oe_1,oe_2,oe_3….oe_n}

n=number of events

V. Related Work

1. We have used aspect oriented programming and reflection to reverse engineering UI's at runtime to translate source platform to target platform..
2. Above mechanism is used to produce the initial UI screen of the target application.
3. Mobile platform provides publicly accessible mapping, in which the API's of the target platform can be used to emulate the functionally of target platform.
4. UIML employs an XML base language to subsequently generate user interface in a desired language.

VI. Proposed Work

We presented a generic approach to execute Android applications natively on the Windows Phone. The ultimate goal of Cloud Twin is to combine the benefits of full source-level cross-platform porting and designing for a particular mobile platform. Future work could extend it in a number of ways to overcome its current shortcomings.

1. Automating Adoption Of API

Cloud twin has ability to automatically update itself to changes source API to target API.

In cloud twin implementation we can see exactly which component android are intermediate language and that intermediate code components in windows phone.

The idea presented in Rosetta & MAM (Mining API Mapping) allows the mapping of different language API. Cloud twin should analyze the difference between the source language and target language.

2. Collaborative Mobile Application

It shows collaboration through the mobile application. It can combine multiple applications into one by connecting multiple source via one unified UI. The UI can be comprised among multiple translated UI by adding page changes in each UI. Then this translated UI could communicate with its own backed source via its own emulator.

3. Supporting Platform

In our implementation we can translate an android app to the windows based apps but with the help of cloud twin. We can translate to other platform also like iOs. For this we need the support of AOP & test automation framework. Using cloud twin we can translate application of android to windows as well as iOs

References

1. E. Holder, E. Shah, M. Davoodi and E. Thievish. Cloud Twin: Native Execution on Android Application on the windows Phone. In ASE 2013.
2. H.Zhong, S.Thummalapenta, T.Xie, L.Zhang, and Q.Wang. Mining API mapping for language migration. In ICSE, 2013.
3. A.Gokhale, V.Ganapathy, and Y.Padmanaban. Inferring Likely Mapping between APIs. In ICSE, 2013.

© 2014, IJARCSMS All Rights Reserved

4. E.Holder, Cloud Twin: Interactive Cross-Platform Replay for Mobile Application. In SPLASH, 2013.

5. Google.Android API reference. http://developer.android.com/reference/packages.html.

© 2014, IJARCSMS All Rights Reserved