A Cross-platform and User Transparent Mobile Application Event Collection Mechanism

Yue Tang [a], Yanlei Shang [b]

Institute of Network Technology Research Center, Beijing University of Posts and Telecommunications, Beijing 100876, China

[a]December_Tang@163.com, [b]shangyl@bupt.edu.cn

Abstract. Nowadays, with the fast improvement of mobile device capability and mobile application development technology, mobile applications with diverse functionalities are gradually becoming an indispensable part of our daily life. When using a mobile application, user's usage behavior could generate some useful information for developers to analyze and improve their applications. However, there are a few ways for developers to particularly acquire application usage information except in a common form of collecting application usage feedback comments from users. In this paper, we propose a cross-platform mobile application event collection mechanism, which could collect application usage events from different mobile platforms. Besides, we use a cross-platform mobile application combined with this mechanism to demonstrate its availability.

Keywords: Event collection; Mobile application; Cross-platform.

1. Introduction

In recent years, smart mobile devices' rapid development leads to application development focus gradually shifting from PC to mobile devices. Mobile applications with diverse functionalities occupy a large part of our daily life. Exceed half number of smart phone owners would use mobile application each day of a month [1], mobile applications with diverse functionalities are becoming an indispensable part of our daily life. When user uses a mobile application, some events would be triggered, such as clicking a button and registering a new app account. All of these events triggered by users could represent users' application usage habit. Sometimes, these events are significant for application developers. They could collect these events to analyze users' behaviors, and make some data-oriented decisions to improve user retention and profitability. However, there are a few ways to help developers to collect detailed application usage information in real time. If developers want to know about their applications usage information in users' mobile devices, they could send questionnaire to users or collect usage feedback comments from some application usage feedback comment systems. But all of these collection ways mentioned above are not intuitive and explicit enough for developers to make correct and effective data-oriented application improvement decisions. After all, not all users would spend time on writing an explicit application usage experience. Although, some developers might write some mobile data collection methods within their applications, these methods are not available for all mobile applications. Thus, when developing an application, developers need a cross-platform and user transparent mobile application event collection mechanism to help them collect user triggered events in real time and submit events to data analytics server for subsequent analytics.

The American Mobile Application Report in September 2015 showed that the American usage rate of mobile application increased 52% compared with last year [2]. This trend leads the development of mobile applications to a great expansion. Nowadays, there are different kinds of mobile application platforms such as Android, iOS and Windows phone being the dominant position. But each of platforms requires separate development approach such as development tool and programming language. These native mobile application development patterns would bring some negative effects [3], due to developers have to solve the inconsistencies between different platforms, and implement different kinds of application versions when developing a same application. But with the help of Web development technologies such as HTML5 and JavaScript, these issues could be addressed since they are independent of platform. With gradually increase demand trends, some

© 2016. The authors - Published by Atlantis Press

cross-platform mobile application development technologies emerge such as PhoneGap, Ionic, etc. These cross-platform technologies are based on Web development manner (HTML5, JavaScript and CSS) [4].

Considering the shortcomings of native mobile application development and lacking an effective application events collection mechanism, in this paper we use a lightweight, event-driven JavaScript framework named Flight along with PhoneGap to realize cross-platform mobile application development. In addition, we design and implement a cross-platform mobile application event collection mechanism which is developed with JavaScript and suitable for cross-platform mobile applications. The main purpose of this paper is to provide the mobile application developers with a cross-platform and user transparent mobile collection mechanism to collect events triggered by users in real time for subsequent analytics.

This paper is structured as follows: Section 2 introduces related work. Section 3 represents the architecture of the cross-platform mobile application event collection mechanism. In Section 4, we provide the details on the implementation of the mechanism. Then, we describe how to combine it with cross-platform mobile application built by PhoneGap and Flight framework to realize mobile event collection in Section 5. Finally, in Section 6 we make a conclusion and discuss future work.

2. Related Work

Nowadays, there are some cross-platform mobile application development tools contributing to solve the problem of repeating programming work for different mobile operating system, such as some very popular tools: PhoneGap, Ionic and Sencha Touch.

Adobe PhoneGap framework is an open source distribution of Cordova [5], which provides developers with a fast mobile application development platform and gives them access to all of the native device APIs such as GPS, Bluetooth, Camera, Contacts, Accelerometer and more, so that the mobile application built with Web technologies behaves just like a native mobile application. PhoneGap is an effective tool for building mobile applications by using Web programming technologies such as HTML, CSS and JavaScript [6]. The way of building application by PhoneGap is called hybrid development, which means PhoneGap applications are not purely native or Web-based [7]. After programming by Web technologies, developers need to export the project to native development tools such as X-Code or Android Studio for for application packaging. Besides, PhoneGap also provides developers the possibility to develop own plug-ins [8].

There is another useful cross-platform mobile application framework called Ionic. Similar with PhoneGap, Ionic provides a powerful HTML5 hybrid mobile app framework, which means developers could build native-feeling mobile applications using Web technologies like HTML, JavaScript and CSS [9]. But different from PhoneGap, Ionic is mainly focused on look, feel and UI interaction with your application. Ionic is a lightweight framework and based on AngularJS in order to work at its full potential. Besides, Ionic offers developers a visual mobile application editor called Ionic Creator. Graggable components and visual mobile emulator bring developer great programming experience.

Flight is a kind of original JavaScript framework, and it has some advantages compared with other JavaScript frameworks. For example, some other web frameworks encourage developers to arrange their code around a prescribed model layer, but Flight is organized around the existing DOM model and its functionality is mapped directly to DOM nodes [10]. The event-driven characteristic of Flight could help developers to monitor application events triggered by users easily, and make use of these events to improve their app.

Nowadays, there are some companies focus on mobile data analytics such as Amazon Mobile Analytics and Alibaba Mobile Data Analytics. These mobile analytics tools contribute to help application developers to collect mobile data to analyze users' usage behaviors, acquire daily active users, average revenue, and app retention and so on.

3. Architecture of Cross-Platform Mobile Application Event Collection Mechanism

In this section, we would introduce the architecture of our cross-platform mobile application event collection mechanism which is shown in Figure 1. And the whole architecture could be divided into two parts, the client side and the server side. This paper is focused on mobile application event collection which is said the client side, so we would describe it in detail. And for the server side, we use an open source data analytics platform named DAS which would be introduced in the second part briefly.

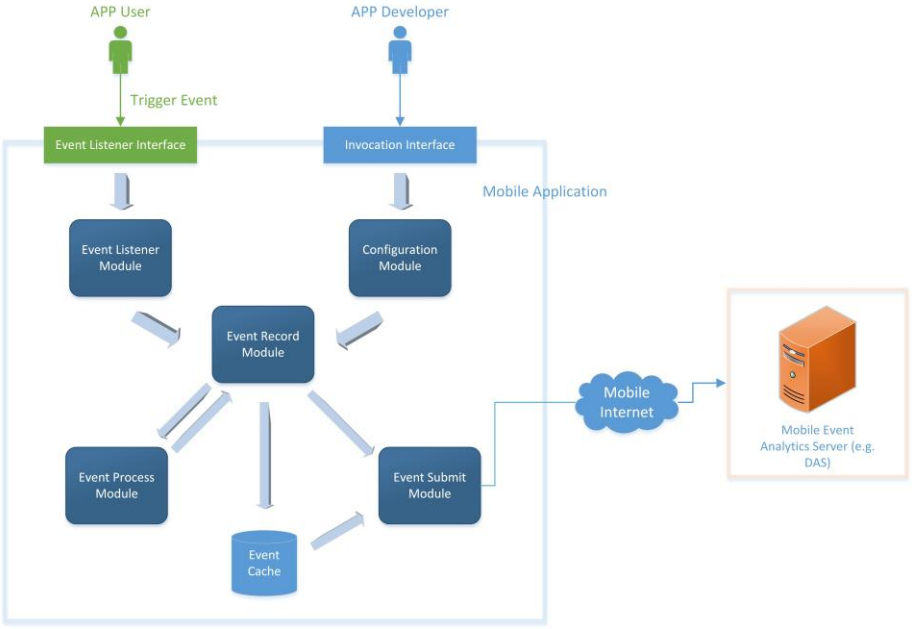

Fig. 1 The architecture of cross-platform mobile application event collection mechanism

3.1 Mobile Application Client Side

There are two roles that would access mobile application, one is App user and another is App developer. Firstly, App user would use mobile app and trigger some events, such as clicking some buttons for click event or registering an account for register event. And these triggered events would be monitored by the Event Listener Module. Then, App developer makes some configuration information in the Configuration Module through the Invocation Interface. After monitoring triggered event along with some predefined event attributes, the Event Record Module would record event, and the Event Process Module would define the format of event by adding some required attributes. There are two ways to submit event, one is pushing event to the Event Cache which is local storage for event cache and another is submitting immediately through Event Submit Module. If developer chooses to push event to the Event Cache firstly, when the quantity of cached event exceeds to predefined max limit, these events would be automatically submitted by the Event Submit Module. And if developer chooses to submit event without caching, event would be submitted to the Mobile Event Analytics Server at some interval (every 10 seconds by default).

3.2 Mobile Event Analytics Server

As for the Mobile Event Analytics Server, developers would program it by themselves, or use an open source product named DAS (Data Analytics Server) provided by WSO2 which is shown in Figure 2. In this paper, we use DAS as our Mobile Event Analytics Server. DAS is a comprehensive enterprise data analytics platform, which could batch and real-time analyze any source of data with predictive analytics via machine learning. And it could support the demands of business, Internet of Things solutions, mobile and Web applications [11]. In a mobile application development context, DAS could effectively help developers to gather useful information, analyze and make well-informed decisions. DAS can collect data streams from data source, run them through real-time, interactive, predictive or batch analytics, and then communicate these through alerts, queries, visualized dashboard or APIs. It's a great data analytics platform that could be combined with our mobile application event collection as an event analytics server.

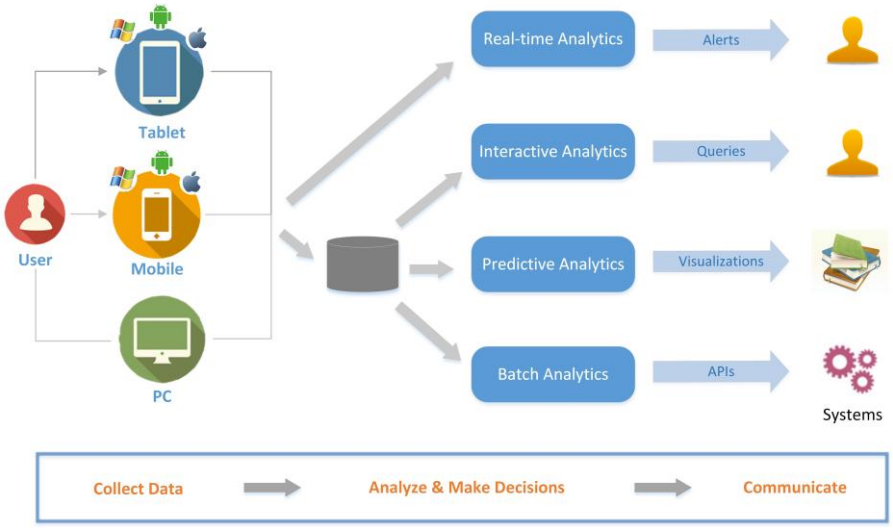

Fig. 2 The framework of Data Analytics Server

4. Implementation of Cross-Platform Mobile Application Event Collection Mechanism

We have introduced in section 3, the mobile application event collection mechanism is combined with several modules: Event Listener Module, Configuration Module, Event Record Module, Event Process Module, Event Submit Module and Event Cache Module. Each of modules has different functionalities, and they communicate with each other via its own defined methods. Figure 3 shows the flow of methods invocation. In this section, we will describe the implementation of each of them in details.

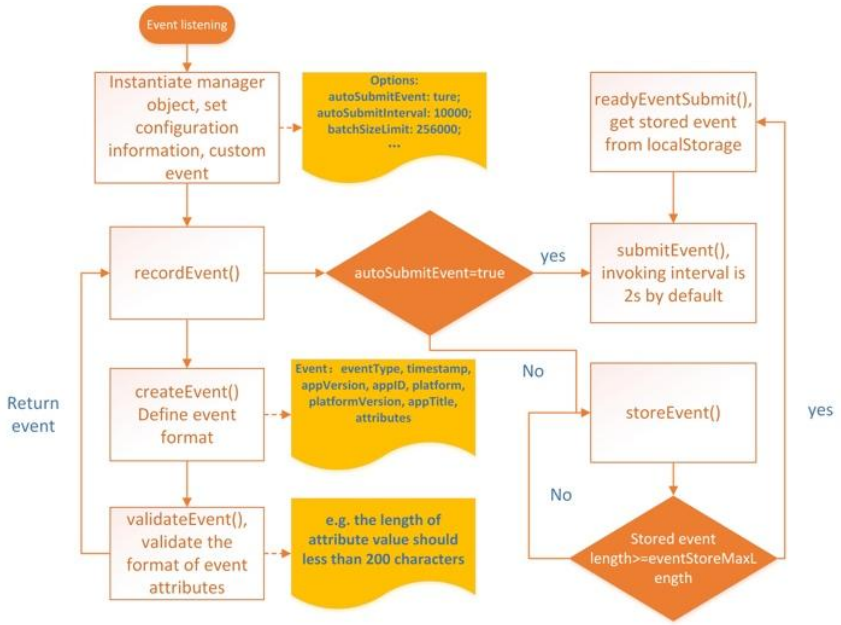

Fig. 3 Methods invocation flow

4.1 Event Listener Module

The main function of this module is to monitor the event triggered by users. Developer could define custom event listeners or use predefined event listeners such as click, submit, touchstart and touchmove events. Because this mobile application event collection mechanism is based on cross-platform, the main coding language we use is JavaScript. Event Listener Module is required in JavaScript file to monitor triggered events. Then, developer needs to bind event listener to required DOM nodes. Developer could define some actions when listening event triggered, for example, we bind this click event listener to a button to record click count.

4.2 Configuration Module

This module is to define some required information and instantiate an event manager object. The required information includes app Id, App Title, app Version Name, platform, platform Version, etc. The Manager object is defined in a JavaScript file, which has some attributes and prototype methods. We implement Manager Class encapsulation via the thought of Object-Oriented programming design, and developer could access these methods as long as instantiating a manager object. The Manager Class's namespace which is a container defines some properties and methods. The idea of creating a namespace in JavaScript is simple: creating a global object, and all variables, methods become the properties of this object. Using namespace also could reduce the chance of name conflict [12].

In the configuration information, appId represent the application ID, which is used to exclusively identify an application if developer has more than one application required to be monitored. AppTitle and appVersionName define the name of application and its version respectively. Platform and platformVersion describe the mobile platform of the application, which could be iOS, Android or Windows phone. AutoSubmitEvents is optional, if developer does not define it, this event collection mechanism would set it as "true" and submit event automatically. Otherwise, event would be stored in Event Cache for later submitting. EventStoreMaxLength is optional too, which sets a limit for event cache. When stored events exceed to this limit, events would be submitted one by one, and at the same time, submitted events would be deleted from Event Cache.

4.3 Event Record Module

This module aims to help developer to record the event monitored by Event Listener Module, and its corresponding method is "recordEvent" method. This method contains three parameters: first is event type which implies the type of event; the second is attributes, which is in a format of key-value pairs and customized by developers; the last parameter is the URL of mobile event analytics server.

Firstly, this module would communicate with Event Process Module to retrieve event final format. Then, it would determine whether sending event to the Event Submit Module for automatically submitting by checking the attribute of "autoSubmitEvents", if it is true, event would be submitted to Event Submit Module, otherwise, event would be stored in Event Cache. When stored event exceed to predefined limit, stored event would be submitted automatically.

4.4 Event Process Module

This module is used to format event, and make some validation. The events prepared sending to mobile analytics server should be unified, which means the submitted event should contain required attributes and be transformed into JSON format. These required attributes are predefined by mobile application event analytics server, which contains the name of attributes such as event type, timestamp, app version, app id, platform, custom attributes, etc. In addition to event format unified, this module is responsible for event validation too. For example, the type of attribute name defined by developer should be "string" and with length shorter than 50 characters. And each attribute's value length should be shorter than 200 characters. If not matching any of these validation items, some error messages would be alerted.

4.5 Event Submit Module

This module is responsible for event submitting, and submitting interval is 2 seconds by default (developer also could define this interval by setting autoSubmitInterval to any time). In this module, we use Ajax HTTP request to communicate with the mobile application event analytics server. AJAX means Asynchronous JavaScript and XML, which allows Web page to be updated asynchronously by changing small amounts of data with the server, and without loading the whole page [13]. In order to realize AJAX, we need get an XMLHttpRequest object, and use its open () and send () methods to send a request to a server. The open () method has three parameters: "method" specifies the type of request (POST or GET), "url" is the server location and the last parameter is Boolean type, true for synchronous and false for synchronous. Although, GET method is simpler and faster than POST, it only supports small amount of data sending from client to server. The request sent to the server contains large amount of event payload, so we choose POST method. Sending asynchronous requests could improve Web performance. Many of tasks executed on the server are very time consuming, this

operation could cause the application to hang or stop if without AJAX. So we set the third parameter as "true" to realize asynchronous request.

4.6 Event Cache

Event Cache is used to store events which do not need to be sent immediately. HTML5 provides two ways realize local storage: sessionStorage and localStorage. The sessionStorage data is stored locally for a session, and when ending the session, the stored data would be destroyed. So sessionStorage is used for temporary local storage. The localStorage is used for persistent local storage, unless the initiative to remove the stored data, or the data would be never out of date. In this paper, we choose localStorage to be the role of Event Cache due to its persistence property.

When Event Record Module finds the event does not need to submit immediately, and it would send event to Event Cache for persistent storage. Event could be stored, got, and removed from Event Cache, and corresponding methods are setItem (key, value), getItem (key) and removeItem (key). The event stored in Event Cache should be in key-value pairs, which is convenient for reading. When the stored events exceed to predefined length limit, these stored event would be submitted and removed from Event Cache

5. Use Mobile Application Event Collection Mechanism With Cross-Platform Mobile Application

Our mobile event collection mechanism is based on JavaScript, so it is applicable for cross-platform mobile applications and Web applications. In this section, we would describe how to combine this mobile application event collection mechanism with cross-platform mobile application to monitor and collect events triggered by users. PhoneGap framework is open-sourced and fast mobile application development platform, and Flight framework is event-based, lightweight and has the advantages of components decouple. So in this paper, we use PhoneGap and combine it with Flight to develop a cross-platform mobile application.

5.1 Build a PhoneGap Project

PhoneGap is a platform for developing cross-platform mobile apps by using HTML, JavaScript and CSS. There are two ways to install PhoneGap: first is using Node Package Manager (npm) to install the PhoneGap CLI, or directly download desktop app. You also could install PhoneGap app for mobile, which will get you running your PhoneGap project on your mobile device without code-signing or compiling. Then, we create a PhoneGap project called FlightPhoneGap. New created PhoneGap project would contain some files and folders, developers would make programming in a folder named "www", which includes HTML, CSS and JavaScript files. And another folder named "platforms" includes the platform files. After coding, we could run our new app on simulator or on a mobile device. Figure 4 shows app running on iphone6 simulator via PhoneGap CLI.

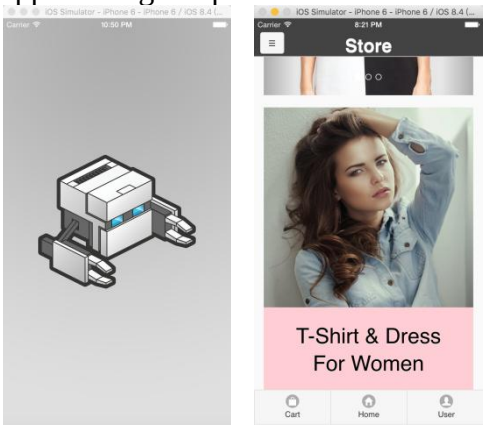

Fig.4 Run app with iOS simulator

5.2 Combine PhoneGap Project with Flight

Flight is a lightweight, component-based JavaScript framework that maps behaviors to DOM nodes. It does not prescribe or provide any particular approach to rending or providing data to a web application, and its components' functionalities are mapped directly to DOM nodes. Flight is

event-based, when a component triggers an event, it has no idea of how its request will be satisfied or by whom. A component is a JavaScript file, which defines its attributes, functions, triggering events, listening events and which DOM nodes its functionality would map to. Algorithm 1 shows the definition of a component. Components do not engage each other directly, they communicate through event broadcasting and event listening, and corresponding APIs are trigger () function and on () function. This enforced decoupling of components, which allows developers to consider each component in isolation rather than worrying about the growing complexity of the application. So we take advantages of Flight to implement our mobile application.

Algorithm 1: component definition

```
var Inbox = flight.component(inbox);
function inbox() {
  this.doSomething = function() { /* ... */ }
  this.doSomethingElse = function() { /* ... */ }
  // after initializing the component
  this.after('initialize', function() {
    this.on('click', this.doSomething);
    this.on('mouseover', this.doSomethingElse);
  });
}
/* Attach the component to a DOM node */
Inbox.attachTo('#inbox');
```

To create a Flight project, we need to install Node and Flight generator firstly. Then, we create a Flight app by the command "yo flight myAppName". New created Flight project would contain many files and folders, and the "bower_components" includes all Flight dependent libraries, such as jQuery and Require.js. Flight uses jQuery and a module loader that supports for AMD, such as Require.js. Finally, we copy this "bower_components" folder to PhoneGap project.

5.3 Use the Cross-Platform Mobile Application Event Collection Mechanism

After finishing a mobile application via PhoneGap, and successfully run it on iOS platform, next step we would demonstrate how to use the cross-platform mobile application event collection mechanism with our mobile App to monitor and collect the events triggered by users. In this paper, we take a button clicking event as an example to show the clicking event monitor and collect via this event collection mechanism.

Firstly, we need to import the cross-platform mobile application event collection mechanism into our PhoneGap project via <script> tag. We instantiate a Manager Object and initialize some configuration information in a component named Search, and attach this component to the search button. Then, we define the click event attributes which contains button id and click count, and use the recordEvent () function to record this button clicking event.

The mobile event analytics server is DAS, before the communication between client and server, we need to configure the payload data attributes in DAS. All attributes name must be in accord with mobile client event collection mechanism, and all of attributes type are "string".

Firstly, we set the value of "autoSubmitEvents" to "true", and click "go" button shown in the red box of the Figure 5. This click event would be submitted to DAS in every two seconds. We use some alert windows to clearly show the flow of event collection. Figure 5 also shows that the mobile event analytics server has received "go" button click event.

Then, we try to set the value of "autoSubmitEvents" to "false", and click "go" button. This click event would be stored in the Event Cache. When the length of stored events exceeds to predefined max length, events would be submitted and removed from the Event Cache. We define the max storage length is 2500, so when exceeding this value, event would be automatically submitted in every 2 seconds.

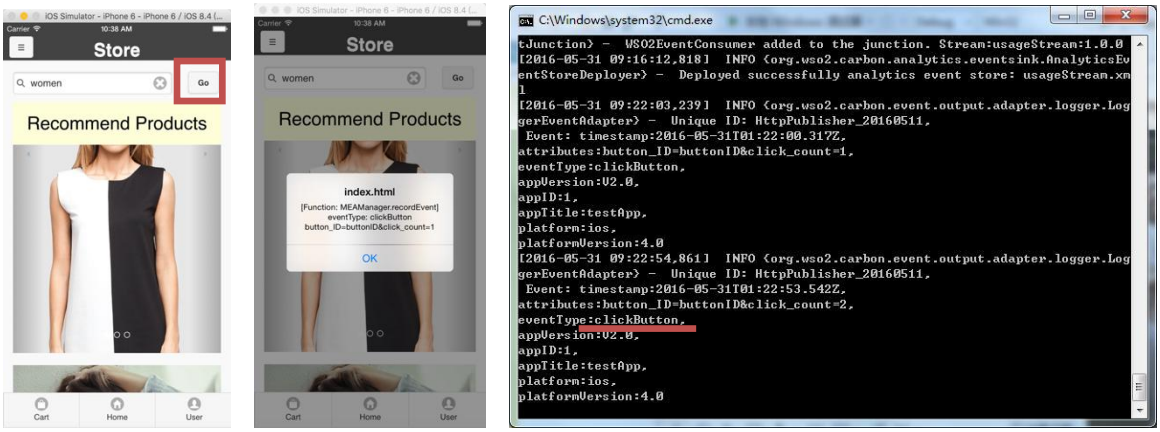

Fig.5 “go” button click event

6. Conclusion and Future Work

With the increased number of mobile application users, application developers need a more effective way to build applications, and a tool to collect the information about users’ usage habits for their applications’ improvement. Cross-platform mobile application development provides developer a fast and effective way to build mobile application for different platforms. Conforming to this cross-platform mobile applications development trend, in this paper we offer a cross-platform mobile event collection mechanism that could help developers collect all events triggered by users and send them to the mobile events analytics server for users’ behaviors analytics. And this event collection mechanism is also applicable for Web applications because of coding via JavaScript. Our cross-platform application events collection mechanism is easy to use, developers only need to import it to application via <script> tag, and customize events collection configurations.

Considering the performance improvement of our application events collection mechanism, in the future work, we would enable a dynamically configurable functionality to the Configuration Module, so that developers could edit application events collection rule. In addition, not only could it send events to the event analytics server, but also could retrieve some analyzed results from server and represent them to users such as some personality recommendation.

References

[1] Information on: http://b2b.toocle.com/detail--6193768.html

[2] Information on: https://www.comscore.com/Insights/Presentations-and-Whitepapers/2015/The-2015-US-Mobile-App-Report

[3] Christos Bouras, Andreas Papazois and Nikolaos Stasinos, “A Framework for Cross-platform Mobile Web Applications Using HTML5,” in 2014 International Conference on Future Internet of Things and Cloud, IEEE, 27-29 Aug. 2014, pp. 420 – 424

[4] Information on: http://b2b.toocle.com/detail--6193768.html

[5] Mahesh Babu R , M. Balaji Kumar, Rakesh Manoharan, M. Somasundaram and S.P.Karthikeyan, "Portability of mobile applications using PhoneGap: A case study”, Software Engineering and Mobile Application Modelling and Development (ICSEMA 2012), International Conference on Chennai, IET, 19-21 Dec. 2012

[6] Information on: http://gigaom.com/2009/04/05/phonegap-seeks-to-bridge-the-gap-between-mobile-app-platforms

[7] V. G. Sarah Allen and L. Lundrigan. PlatformDevelopment. Apress, 2010.

[8] Manuel Palmieri, Inderjeet Singh and Antonio Cicchetti, “Comparison of Cross-Platform Mobile Development Tools,” in 2012 16th International Conference on Intelligence in Next Generation Networks, IEEE, 8-11 Oct. 2012, pp. 179 – 186

[9] Information on： http://ionicframework.com/docs/overview/

[10] Information on: http://flightjs.github.io

[11] Information on: http://wso2.com/products/data-analytics-server/

[12] Information on: https://developer.mozilla.org/en-US/docs/Web/JavaScript/Introduction_to_Object-Oriented_JavaScript

[13] Information on: http://www.w3schools.com/ajax/ajax_xmlhttprequest_create.asp