International Research Journal of Engineering and Technology (IRJET) e-ISSN: 2395 -0056
Volume: 04 Issue: 02 | Feb -2017 www.irjet.net p-ISSN: 2395-0072

Automatized Application Services for Android Devices Using Apache Ant in Cloud Environment

Priyanka.V[1], Sushmitha.B.S[2] and Rupa Kesavan[3], Kapila Vani. R. K[4]

[1, 2] Department *of Computer Science and Engineering, Prince Shri Venkateshwara Padmavathy Engineering College, Chennai, India.*

[3]*Assistant Professor, Department of Computer Science and Engineering, Prince Shri Venkateshwara Padmavathy Engineering College, Chennai, India*

[4]*Assistant Professor, Department of Computer Science and Engineering, Prince Dr. K.Vasudevan College of Engineering and Technology, Chennai, India.*

----------***----------

Abstract: *The mobile users and the mobile applications are increasing rapidly. Handling different application services individually under some sequence for particular purpose is inconvenient for the android users. To bring it to an end, the mobile end users can able to customize their application services dynamically in a single mobile application based on Model driven pattern. At present mobile offers service functionalities in mobile devices through Native applications, Services available on Web sites and the application that integrate the native functionalities with predefined services. The end user needs to approach the developer for developing these kinds of application with different requirement services. Instead, Micro-App can be developed automatically in the cloud which is the integration of the application services by using Ant tool. The end user customizes their application services via the Micro-App and deploys it in mobiles based on the service requirement pattern.*

Index terms: **Mobile Application Services, Micro-app, Keystore, Ant Build tool, App–Generator Framework, Dyna-Mapper algorithm, Android devices.**

1 .INTRODUCTION

Today's mobile environment are very drastic in terms of performance enabling the new different breed of application to be developed and makes the existing applications to be migrated and used in the mobile platform, which increases the user experience.

The huge development of mobile devices has raised the improvements in the mobile applications. Already the number of mobile devices exceeded the people population in the world. Some applications include integration of many application services that are already available on the mobile devices. For example, some application services in mobile devices are the camera, contact list, device status, weather, standard encryption and standard decryption, etc., Mobile End user uses these application services separately or as an application which integrates some application services. The application which has the integration of the application services are already developed and hence end user can only able to deploy it. There are no possible ways for changing some requirement in the already existed application.

If the end user requires any changes in the application services, the only possible way is to approach a developer of an organization to make those changes. Also if the user needs any new application with a new set of combination or integration of application services, approaching the developers are the only way. Here the user provides the requirement details of the application services to the developer. The developer obtains the entire requirement from the user to startup with the application development which includes all the application services. The developer uses the IDE's (Integrated Development Environment) such as eclipse, Android Studio, etc., for the application development. Some issue in this way of developing an application includes more time in the development of application services, impossible to change application services dynamically and also expensive. On the other hand, in some cases, the user also requires some technical knowledge and programming skill about the services.

To efficiently use the application services available in the mobile devices, a Micro-App can be generated. This Micro-App is an integration of the application services that are available on the mobile devices. The Micro-App can be generated without using any IDE's. The mobile end user can provide their requirements of services through some application user interface (UI) by using their mobile devices to the service provider. Ant build tool which is to build the Application. It is java build tool which automates the build process of the application. On the mobile end user request the Micro-App application can be built automatically using this Ant build tool. The Micro-App is authorized by the keystore files only then the user can deploy this application on their mobile device. The end user can also customize the application services in the Micro-App application.

The proposed paper is structured as follows. Section II reviews on the related works. Section III describes the work

carried out to customize the application services. Section IV includes the generation of keystore files. Section V mentions about the Dyna-Mapper algorithm and Section VI gives the framework of proposed system. Section VII describes the process in developing the Micro-App. Section VIII is the conclusion and future enhancement.

II. RELATED WORK

There are many existing mobile application services (camera, video, email, encryption and decryption, etc). However, the integration of those services can be done by the developers. But we integrate those services automatically with the help of [1] Apache Ant based on the different user requirements

A brief history of existing mobile application services:

i. *Native applications*: These are the applications which are developed, compiled, installed and deployed on mobile devices natively. Native mobile applications are built using framework such as Objective-c framework in iOS and JAVA/ Android SDK framework in Android. These application services include contacts message, camera, etc.

ii. *Services that available on websites*: The service consumer and the provider use messages to exchange invocation request and response information in the form of self-containing documents. The Big and the RESTful web services are the commonly used services. The Big uses the XML messages that follow the Simple Object Access Protocol (SOAP) standard, a language defining message architecture and message format. The Representational State Transfer (RESTful) is integrated with HTTP than SOAP-based services, do not require XML messages or WSDL service-API definitions.

iii. *Applications that integrate native functionalities with predefined services:* The application that is built with available predefined services. For example, a camera application that enables to post a photo on the Face book profile.

Apache ant [9] is a build tool in java which provides platform independence and immediate integration to a newly adopted system. This project demonstrates how the workflow is being managed based on the interactive tools of Ant. The Ant starts building the application based on the commands that is provided through the command line. This building reduces the work of developer as well as it allows the user to use their customized services based on their own requirements. At this building stage, the Ant integrates the application services similar to that of Data integration [2] which is a framework to unify both data and the procedures.

The Ant fails to authenticate the user. This can be done here with the help of Keystore files [3]. Keystore is a storage facility for cryptographic key and certificates. Keytool is a key management tool that enables the user to administrate own public/private key pairs and certificates for use in-self authentication. It also maintains the cryptographic keys and certificates in a keystore. The generation of keystore is done based on the RSA algorithm. The emerging technologies such as Internet Of Things and Cloud Computing are integrated and used as Cloud of Things(COT) [4] in which the cloud based IOT accumulates Iaas, Saas, Paas for accelerating IOT application, development and management. Saas (Software as a service) is used here in the cloud to convert the source files into Apk files with the help of Ant. Model Driven Architecture (MDA), which is the enhanced approach [5] for developing an application with different user requirements (needs).Model Driven Architecture initiated by Object Management Group (OMG) makes the people to find the content easily. Based on the pattern the users customize their application services, which can be used in a specific order or an independent application.

Zhenlian Peng[6] presented the concept on the issues of service variability. Service variability refers to the changes in the requirement of different users. The concept is about the model which deals with the end user with different application service requirement. The model allows abstracting the variability of services to customize the services. It also validates and authorizes the customization of services and also provides the dynamic deployment of application services.

The concept of the service based mobile application (SMA) [7] overcomes the drawback of limited resources for the mobile application to be developed. A computing platform called Super Mobile Autonomous Reliable platform (SMART) used in SMA supports the development, distribution of Mobile Applications, Development, Deployment and Management of cloud services, Management of Services Subscription. Through this concept high performance, high portability, high manageability, easiness of design and developments of services and mobile applications are achieved. Software build systems [8] are complex which involve both static and dynamic development environment. It's responsible is to convert the source code into executable libraries. ANT (Another Neat Tool) build develops the software automatically in the dynamic environment.

III. CUSTOMIZATION OF APPLICATION SERVICES

There are many ideas and technologies for end users to customize their mobile applications. Here the user customizes the services via the initial application which we developed and is to be installed in the mobile. All the default services are grouped and made available for use with the micro-app in multiple combinations. After a valid login, user redirected to service layout for a list of available services in a smart phone. The user has to customize it based on their requirements by dragging and dropping the services. The requirement may differ from every user and they can

dynamically alter their requirements, even after the final application development. The initial end user application involves the options such as available combinations (available services), check combinations, scanQR (to download the integrated services as a single application), logout. Nearly 80 possible combinations are available. The following figure 1 shows the selection of the services by the mobile end user.

Figure 1: Selection of services

Some list of available combination of services:

i. TP,PP,SE,GPSLoc,WInfo,CL,SMS,Email.

ii. TP,PP,SE,SD,SMS,EMAIL.

iii. TP,PP,GPSLoc,SE,Email.

iv. Email,SD,SMS.

Code to Drag and Drop the Application Services:

1. *Interfaces:*

The following interfaces and their methods are used to select the application service by dragging particular service icon and to drop it in another selection area.

A. OnTouchListener actions:

i. ACTION_DOWN

B. OnDragListener actions:

i. ACTION_DRAG_STARTED

ii. ACTION_DRAG_ENTERED

iii. ACTION_DRAG_EXITED

iv. ACTION_DROP

2. *Managing service list in array*

The selected application services can be maintained in an array list. The list is in an order so that the service on the final application will follow the same order. Services are assigned with the id for the easy access of the application services. This array list is sent to the server for the development of the integrated application using the Ant build tool.

Services	*Abbreviation*
DS	Device Status
VP	Video Preview
TP	Take photo
PP	Photo Preview
SD	Standard Decryption
GPSLoc	GPS Location
WInfo	Weather Information
RV	Record Video
SMS	Shor1t Message Service
CL	Contact List
Email	Electronic mail

Table 1: Service list

IV. GENERATION OF KEYSTORE FILES

Keystore is a container that stores the cryptographic key and certificates. The main purpose of keystore is to convert unsigned apk into signed apk file. Normally in building android application these conversions are automatically done by generating debugging keys. These keys expire and need to be regenerated periodically.

Ant Build tool does not generate the debugging keys. Instead, a keystore is generated and maintained separately which are permanent to an application (Micro-App) to which it is assigned when requested for the development. The keystore is used to authenticate the final Micro-App. This provides security that prevents, various things, remote attackers from publishing malicious software or updates to that final application. This keystore is managed by the keytool. In general keystore signs the apk file of the application. The server generates the keystore and stores those keystore in the database. For this the server sign in as admin and the keystore can be generated by using the buttons. On successful creation of the keystore, the command prompt on the server automatically invoke where the additional credentials for that particular keystore can be given. These credentials are for the security purpose and the keystore is also used to track the application for which it is assigned.

V. DYNA-MAPPER ALGORITHM

On the server end, while the request to build the Micro-App is received, the server starts to search and add the necessary files from the server database. Service is received with the array list of services using which the files are searched. Server contains layout, jar and java file for all the application services in its database. It checks the array list iteratively and search the files required in the database dynamically. It is an algorithm that works dynamically on different services integration request. Each java file of the application services has a method which decides which service should be developed or deployed next. The algorithm is as follows

Algorithm: Dyna-Mapper Algorithm

Input: List of services from the mobile end user.

Output: Package with all the application services files to build the Micro-App apk file.

function dyna-mapper (array list)

{

Step 1: for each service i in the array list do the following step 2 and step3.

Step 2: check each service with all the services available at the server end to find the match.

Step 3: If match occurs, get all the java, jar files of that particular application services.

Step 3.1: Every java file of the service has a procedure through which the dynamic integration of the services occurs. This procedure decides the next service to be integrated with the reference of the input array list.

Step 3.2: While deploying the services, the next service is invoked from this procedure of the existing service.

Step 4: The files obtained are combined into the package. Finally return the package

}

The package from this algorithm is the source code for the Ant build tool to develop the apk file. For the layout design, the jar and layout file are combined and these files are also given to the Ant build tool. The following figure 2 describes the entire process of developing the integration of services in a single Micro-App application.

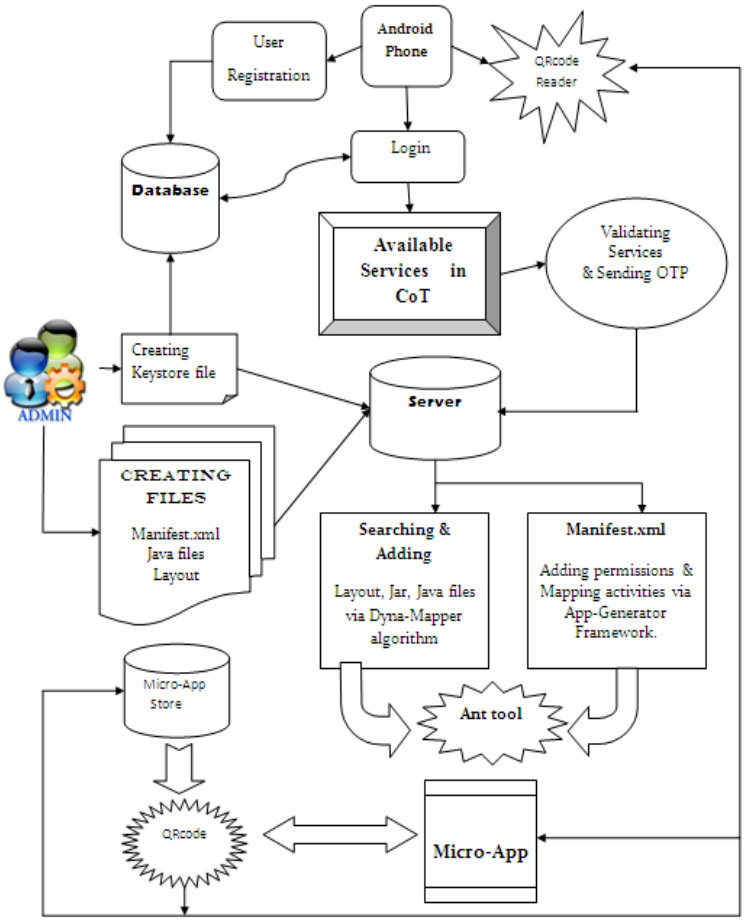

Figure 2: Process in developing Micro-App application.

VI. APP-GENERATOR FRAMEWORK

App-Generator is a one which generates the android manifest file automatically. Android manifest file is an xml file that provides essential information about the application, which the android system (mobile phones) must have before it can run any of the application's code. Manifest file describes the components of the application, which include the activities, broadcast receivers, services and content providers that compose the Micro-App application. It also declares the permissions that the Micro-App must have in order to access protected parts of the API and interact with other applications. It also declares the permissions that others are required to have in order to interact with the application's components.

The manifest file created by the App-Generator provides the following permission to the application services and these are prefixed with android.permission.

i. Camera

ii. Read_internal_storage

iii. Read_internal_storage

iv. Access_network_state

v. Send_sms

vi. Receive_sms

vii. Access_fine_location

viii. Access_coarse_location

ix. Access_location_extra_commands

x. Read_phone_state

xi. Internet

xii. Record_audio

VII. BUILDING APK FILE IN ANT

After the successful creation of the package file (integration of application services) by App-generator Framework and manifest file, the final Micro-App can be built in the Ant built tool. Apache Ant is a java and command line tool which builds the application automatically. Ant tool has a number of built in tasks allowing to compile, assemble, test and run both java and non-java applications. Ant tool get the input as the package from the server to build the application. With that input, the Ant build generates the apk for the final Micro-App using the Ant commands.

Server initiates the Ant tool to build the application. Command "android update project -t android-10 -p path -n Apk-Upload-File-Name" can be used for building source (an xml file). Command "ant release" and "pause" are used to build the apk file. This is the apk file of the final Micro-App requested by the user which has the integration of application services.

The generated apk file is stored either in the cloud where the ant build tool is deployed or on the server database. When the user is required with the final application, they can download the final Micro-App from their initial application where they gave the request for the integration of services. To download the Micro-App the user can visit the web end and sign-in as user using mail id and password. There, the user requested applications are listed along with the Qrcode. The Qrcode contains the path where the Micro-App is stored. It may be either the server database or the cloud space. By scanning the Qrcode the user can download the application and deploy it. The following figure 3 shows the deployment of the Micro-App (integrated application services) by the mobile end users

Figure 3: Deployment of Micro-App

The user can also able to change the order of application services whenever required without rebuilding the application again. Thus the user can able to use all the application services available in the android smart phones in a single application developed using the Apache Ant Build tool in cloud.

Advantages of using this application:

1. Different end users can customize the application services with different requirements.

2. No programming skills required for users to develop application.

3. Cost effectiveness.

4. Developing android applications requires less time.

5. End user can adjust their behaviour to current context.

6. A single user can customize the application services as many times based on their requirements.

VIII.CONCLUSION AND FUTURE ENHANCEMENTS

Our aim is to provide the customized application services to the mobile end user by integrating the functionalities and services available on the device as a single application (Micro-App) with the help of Ant build. In IDE's such as Eclipse, Android studio, etc., the developer needs to recode everything from scratch if any changes

occur, whereas the Ant facilitate the developer to just update the code where the changes required. Usually, debugging keys are automatically generated as a background process in IDEs (Android studio and Eclipse).These keys expire and need to regenerate it periodically that leads to duplicate key assignment and lack of user control. The Ant Build does not generate these keys. Instead, keystore is generated and maintained separately which are permanent to an application (Micro-App) to which it is assigned.

When different mobile end users approach to develop the application for integration of same services, our proposal builds the same application many times for the requested users. Future research will focus on building the same application only once and enabling the different user with the same requirement to make use of it. And also the Micro-App can enable the Android user to request the required additional services. This integration of additional services on user demand also enhances the usage of Micro-App application.

REFERENCES

[1]Ahrnetsayar,"ant based interactive tools for workflow management" .8th international symposium on intelligent systems and information,september,2010.

[2]n. Lasierra, et al., "an autonomic ontology-based approach to manageinformation in home-based scenarios: from theory to practice", data &knowledge engineering, (2013).

[3] Alessandro cilardo,Luigi Romano"an FPGA-based key-store for improving the dependability of security services"

[4]Zhou, j., Leppanen, t., Harjula, e., Ylianttila, m., ojala, t., yu, c., &jin, h. Cloudthings:"a common architecture for integrating the internet of things with cloud computing", 2013 ieee 17th international conference on computer supported cooperative work in design.

[5]Mukhtar, m.a.o., Hassan, m.f.b., jaafar, j.b., rahim, l.a.:"enhanced approach for developing web applications using model driven architecture". International conference on research and innovation in information systems (2013).

[6] Zhenlian Peng1,2, Jian Wang1, Keqing He1, Mingdong Tang"Web Service Customization based on Service Feature Model", 2015 IEEE International Conference on Services Computing.

[7] "SMART: A Platform for Developing and ManagingService-Based Mobile Applications" Du Wan Cheun, Hyun Jung La, Sang Hun Oh, and Soo Dong Kim.

[8]"TheEvolutionofANTBuildSystems" Shane McIntosh, Bram Adams and Ahmed E. Hassan.

[9]"Ant: Automating the process of Building Applications" N Serrano, 2004.

[10]Huang, W., Wei, X., Zhao, Y., Wang, Z., and Xiao, Y.: "A Multi-tenant Software as a Service Model for Large Organization". 2013International Conference on Cloud and Service Computing, pp. 112-119 (2013)