A New Data Acquisition Client-software Model Used for Mobile Application Analysis

Xiaodong Liu, Zhiyi Qu, Ruixiao Zhang

School of Information Science & Engineering, Lanzhou University, Lanzhou, China

*liuxd14@lzu.edu.cn, quzy@lzu.edu.cn, zhangrx2014@lzu.edu.cn

Keywords: data acquisition; software model; Mobile Application Analysis; Android

Abstract. This paper proposes a new data acquisition client-software model used for the Mobile Application Analysis. Compared with the existing software models, the model proposed in this paper can improve the reuse rate and reduce the code duplication rate of software, it can also share data between multiple applications. The paper gives overview of the software model, then discusses the necessity and function of each module respectively and defines the call relationships between modules. To explain the runtime status of the model accurately, this paper also defines internal and external event types that drive the model. In order to verify the feasibility of the proposed model, the experiments are performed on the Android operating system using the proposed model. At last, the experimental results show that the proposed software model is feasible.

1 Introduction

With the rapid development of information technology, the enterprises are inseparable from the network to survive and develop in the new information era [5]. The network also brings hidden danger in communication between enterprises. In 1988, network identification function is provided by ISO framework, which is called X.509 protocol. The most important part of this protocol is public key certificate. In this paper, a digital signature scheme based on digital certificate is used to ensure integrity of online trading data, non-repudiation of transmitting information and certainty of traders. Global mobile application market is very prosperous with the fast development of the Mobile Internet and the explosive growth of mobile terminal device. Prosperity of mobile application market also makes the competitions among application developers. In order to occupy a place in the fierce competition in the market, mobile application developers must do the application analysis. Mobile application of analysis means to obtain basic data of mobile users to use applications, use the theory and techniques of the Big Data, dig user's characteristics, identify gaps in product design, discovery operation promotion opportunities, and optimize product and operational strategies to enhance service quality.

As data acquisition is the first step in the Mobile Application Analysis, it is very important for subsequence processing. In recent years, there are some research results on it, She’s work [1] presented a data acquisition component, this component is used as SDK to embed into applications. Kang and Qu proposed a data acquisition client software that implemented with Ajax. The component proposed in [1] implements some general functions, but its reusability is poor, besides, its code duplication rate is high. The client proposed in [2] is a lightweight approach, but the data sending frequency is too high and the lack of consideration for the cloud response to client timely. This paper improves the model proposed in [1], and put forward a new mobile application analysis data acquisition client software model that can remedy the defects existing in [1, 2] and bring new capabilities , as shown in Figure 1.

2 Software Model

The main idea of model proposed in this paper is to extract general functions of the mobile data collection as Server and keep all the functions that are directly related to the data collection as Client.

Therefore, the data acquisition client model can be divided into two parts: (1) Client, the Client is alive in a process of application essentially, the user to use this software model define data acquisition method and embed into applications. In addition, Client needs to interact with Server. (2) Server, Server is a system service process, provides cache management, data encryption and other service for each Client. At the same time, Server is also responsible for dynamic configuration and data transmission between whole data acquisition model and cloud.

Client and Server are in a different process, Client need to send a request to the Server to call the service. In order to adapt to the needs of different tasks to send the request, the client can send synchronous and asynchronous request. When Client sends a synchronization request, the Server response is immediate and it seems that all operations complete in one process from the external point of view. When Client sends an asynchronous request, the Client does not wait for the Server to perform the task and return the result. The Client defines the interface to respond to the Server callback. Therefore, Client needs to define the interface to the Server request and implement the Server asynchronous callback method. While the Server needs to implement the Client request interface. If an asynchronous request interface is implemented, the callback implemented in the client is called by Server.

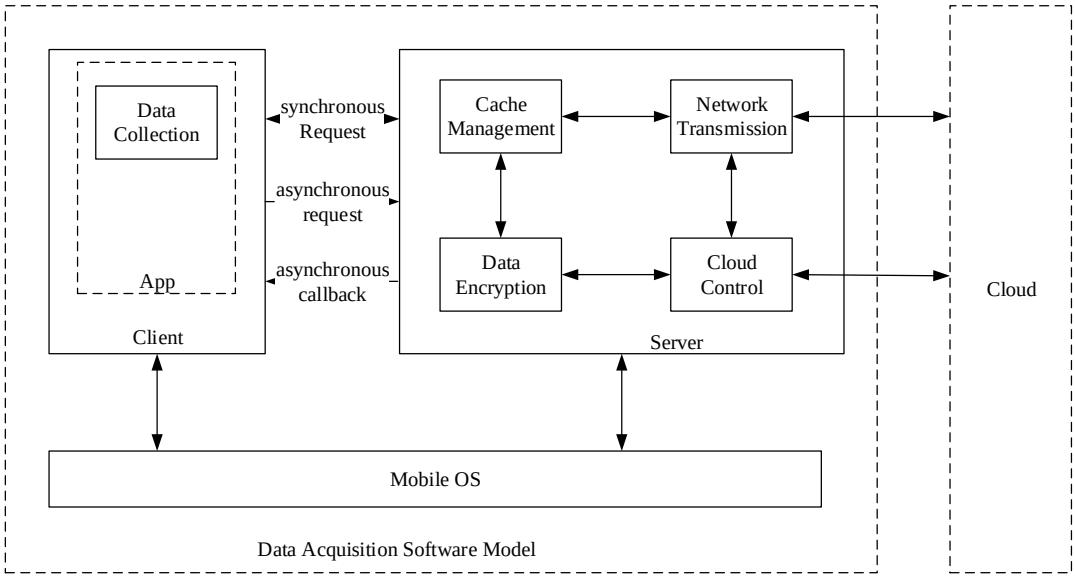

Figure 1 Data acquisition software model

2.1 Data Collection

Mobile Applications Analysis data objects [2] generally include: application distribution channels, the audience attributes, user behavior and terminal equipment. In the process of data collection, the application distribution channels, audience attributes and terminal equipment and other data acquisition targets are implicitly collected, but user behavior is explicitly collected. The implicit collected data is mostly static and conversely and the explicit collected data is relatively dynamic. The function of this module to be achieved should be the union of them, then we focus on explicit collection.

Explicit data acquisition is defined as a collection of methods. These methods can be divided into these categories [4]: basic statistical functions, custom event statistics, error statistics, and social statistics.

(1) Basic statistical functions includes account statistics, page statistics. For example, the time users stay on one interface is a content page statistics.

(2) Custom event statistics is an extension mechanism that allows you to define event ID, parameter names and values to extend the statistics, but this custom events often have to rely on the

basic statistical event. Custom events can be divided into the count events and the calculate events. The count events count the number of times a particular event occurs and the calculate events count distribution of numerical values of variables.

(3) Error statistics major recording application crash logs.

(4) Social statistics are a variety of social behavior in the detailed statistics applications and after further analysis and processing it will form a rich social analysis report.

In order to meet the needs of different user data acquisition, users need to develop their own methods based on the existing data acquisition methods or directly develop new methods to replace the existing methods. Therefore, in the data acquisition module can achieve a certain amount of software reuse and reduce the amount of code duplication. In addition, all the common functions of Server are extracted to form a whole, which can also improve software reusability and reduce code duplication.

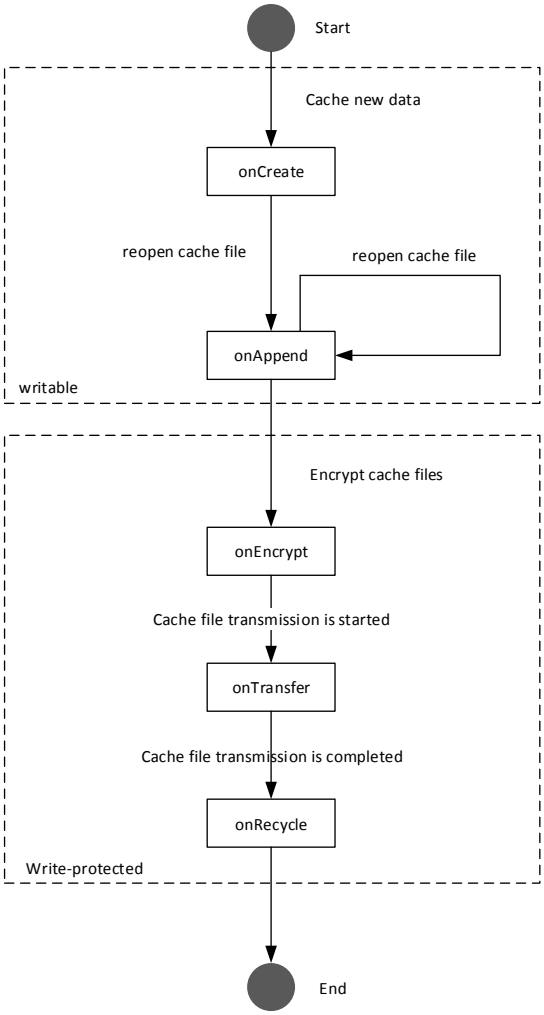

Figure 2 Cache files status

2.2 Cache Management

The necessity of caching: (1) the instability of the network state may result in incomplete data transmission; (2) sending the collected data in the WIFI state can save the data traffic for the user.

In order to get through the local application data, this paper proposes a component model, which requires centralized cache data. In order to collect data from the centralized cache, it is necessary to establish a stand-alone caching warehouse. In the view of static structure, cache warehouse includes two parts: a part is a large number of cache files, the cache file storage various types of data from Client; another part is called cache files state table, which is used to record cache files state (Figure 2), when changing on the state of the cache file, cache file status table also changed Correspondingly.

Cache management is the running state of the cache repository. The cache management need to keep track of each state of the file cache, monitor the cache file life cycle (Figure 2), delete junk cache files and reduce the waste of storage space. Cache management needs to respond to the client side of the request interface for each Client to provide the available cache files, but also to ensure that the outer (including the client) access the cache file legally. After receiving the legal data transmission requests, the network transmission module calls the interface of the cache management to obtain the data to be sent, and transfers the given data.

Cache files are managed centrally in this paper. The statistical data can be shared by applications which use the proposed model in a mobile device. Shared data are generally static, such as the distribution channels, audience attributes and terminal equipment and other data acquisition targets. Therefore, the model proposed in this paper can push the compound data of multiple applications into cloud, and improve the value of the data collected.

2.3 Data Encryption

In order to ensure the security of the collected data in the network transmission, the model of the collected data is encrypted. In addition, there are certain corresponding relationship between the cloud decryption algorithm and the client of the encryption algorithm, and in order to improve software modifiability, so the encryption function will be independent module. When the cloud decryption algorithm changed, the client model only need to modify and replace this module.

The cache management module calls this module to encrypt the cache file. Before running the encryption algorithm, the module needs to call the function of the cloud control module to obtain the latest encryption key.

2.4 Network Transmission

Transmitting collected data completely and saving the user data traffic need to detect the current network stability and connection types, respectively. In order to decouple the relationship between the data file, the transport address and the specific network protocol, so the network transmitting function is independent when constructing the model.

Network transmission module needs to assess the current state of the network, when the network state does not meet certain conditions, the server can refuse the client to send data requests. This condition can be the type of the network connection, signal strength, bandwidth, or the combinations of them. The network transmission module uses the network protocol (HTTP/HTTPS) to transmit data, and the choice of the specific protocol also needs to be agreed with the cloud. When the data files that need to be transmitted are beyond the maximum capacity supported by the network protocol, the network transmission module transmits the data files by block. In order to improve the performance of the transmission, the model suggests using multiple threads to transmit data.

The network transmission module needs to call the cache management module to obtain the data file, which calls for the cloud control module to obtain the address of the data file transfer and the type of the network protocol.

2.5 Cloud Control

When the model is created, this module is added to meet the needs of the dynamic configuration of the client and to reduce the coupling.

There are two ways to implement this module: (1) pulling mechanism, cloud control module holds a fixed cloud request address, select a time a request to the cloud server, query the server settings, so that you can dynamically change the data transmission address; (2) pushing mechanism, the client and the cloud to maintain a long connection (socket), configuration information through long links directly pushed to the client. This method can also meet the need of dynamic information configuration.

However to implement this module, the module must provide network transmission module with interface that is used to query the type of network protocol and destination URL, and provide data encryption module with encryption algorithm and encryption key. In addition, this module must ensure that the client configuration information is the latest.

2.6 Cloud Interface

As can be seen from Figure 3, the cloud needs to provide the client with two interfaces: (1) the cloud control module obtain dynamic configuration from the cloud; (2) network transmission module transmit files to cloud. The former needs to maintain the security of the communication and the connection of the fixed. The latter requires high performance and data file recovery to be maintained.

3 Internal and External Events Running on the Driven Model

In this paper, the main event sources of the client model include: the use of user to the application, dynamic configuration of the cloud and the connection established between Client and Server. For a more detailed specific acts described in the event-driven model, the paper defines the external drive events list, as shown in Table 1. In order to distinguish the nature of the event, 7 conceptual event types are proposed. Some event types are defined specifically for their specificity and importance. For example, the *appOnLoaded* is used to refer to a user initiated application. The *appOnEmbed* event type, because of its complexity is defined as a more generalized abstract event type. These events are an important part of the interaction between the system and the external system. It is very important to improve the usability of the system.

Event Type	Description
appOnLoaded	*Users start the mobile application with analysis function, the Client send a connection request to the Server.*
serverOnBind	*Server verify the legitimacy of the Client. If valid, the connection object is returned. Otherwise, the connection is rejected.*
clientOnConnected	*The Client accepts the connection object, requests the Server for cache resources or requests to transmit the data collected last time.*
appOnQuit	*Normal exit, the Client turn off the connection to the Server.*
appOnError	*App crashes, the Server records exception logs.*
appOnEmbed	*The method of data collection is achieved through embedding in the application or event interception. Therefore here defines an abstract type of event. Perform the appropriate methods of data collection in these events.*
cloudConfigChanged	*Events are triggered when the cloud configuration changes.*

Table 1 Event type description

4 Verify the Feasibility of the Model

4.1 Functional Verification

The client software model covered in this article is based on the literature [1] derived from the past. Compared with the literature [1], the difference is that this paper abstracts more common functions in the whole process of the mobile app data collection to form a single execution Server, and all the functions directly related to data acquisition are reserved to Client. Therefore, this paper verified the structure of Client/Server from the aspects of the functionality and performance.

Service is an application component provided to the application developers by the Android system. Since its lifecycle is particular, it is still in the background when users switch applications. Service can be bound with other application components, communicate with the IBinder.

IBinder [5] is a remote object interface, the kernel is a remote procedure call mechanism, which is designed for high performance process and inter or outer process calls [5]. AIDL (Android Interface Definition Language) is the interface description language provided by Android system. Here, AIDL is a form of IBinder. Through the interface defined by the AIDL, the user in the program to achieve the interface and then the defined IBinder is obtained.

In order to verify the feasibility of Client/Server, this paper implements a software prototype for system simulation data acquisition based on Android system. Server is implemented by using Service,

and Client is implemented by using Activity since it is embedded in the application. Server and Client are running in different applications (or processes). The communication between Server and Client is realized through IBinder, which involves multiple threads, and this paper chose AIDL to implement IBinder.

In order to verify the functionality of synchronous or asynchronous request sent from the Client to the Server, this paper implemented the AIDL interfaces as shown in Table 2. By programming and testing, we can know that both synchronous and asynchronous request functions can be implemented.

Interface Name	Description	Execution Mode
Synchronized-Request	*Simulate the synchronization request, return the system time of the request start, which is used to calculate the response time of the Server. This interface is implemented at Server end.*	*Synchronous*
Asynchronized-Request	*To simulate the asynchronous request, the system time that Client sends the asynchronous request as the input parameter, which is implemented at Server end. Asynchronous tasks use a new thread to start and sleep a second to simulate. After the task completes, the asynchronized callback is called back to calculate asynchronous delay of the task.*	*Asynchronous*
Asynchronized-Callback	*Callback in Server end, and the delay time of the asynchronous task is calculated in Client.*	*call-back*

Table 2 AIDL interfaces

4.2 Performance Verification

In order to verify the performance of Client/Server structure that can meet the need of application statistics, this paper designed a testing process which is used to test the response time of Server synchronization request and the asynchronous delay time of asynchronous request. The response time of the synchronization request is the time interval of the Client in which the request is sent to the Server to execute the request. The delay time of the asynchronous request is the time that the Client is required to perform asynchronous tasks when the asynchronous request is sent to the Server to complete the asynchronous task callback start time.

Before the experiment we need to declare that: (1) according to the latest report released by the TalkingData, the national average each mobile has installed 34 applications. So the Client number interval is set to 2-27 in this paper; (2) the asynchronous task performed by open threads to sleep 1 second way to simulate; (3) under normal circumstances, only one application is running in the foreground of mobile devices. Therefore, each Client works out the synchronous and asynchronous request one by one.

Definition of the testing process: (1) start the specific number (2-2^7) Activity (Client) and connect to the Service (Server); (2) per client implements one after the other 100 times Table 2 which shows the synchronous or asynchronous request and synchronous or asynchronous request by random number to 2 to take the remainder to determine; (3) to calculate every time synchronous request response synchronization time and asynchronous requests asynchronous delay time; (4) to determine the number of clients, to calculate synchronous average response time and average asynchronous delay time. The results of test are shown in Table 3.

CN	SIN	AIN	ART（/ns）	AAD（/ns）
2	101	99	147860	3708900
4	211	189	192576	4765348
8	405	395	187232	4195099
16	799	801	164638	5620149
32	1598	1602	154710	5535376
64	3193	3207	146748	5640941
128	6451	6349	153386	6727544

Table 3 CN, Clients Number; SIN, Synchronous Invoke Number; AIN, Asynchronous Invocation Number; ART, Average Response Time; AAD, Average Asynchronous Delay

We can be seen from the table 3, the response time of the synchronization request is much less than the 5 second which is the limit of Android UI ANR (Not Responding Application), therefore, the application of Client embedded will not affect the operation of the application. The delay time of the asynchronous request is much less than the time of the actual execution task, so that the Server can handle the client's time consuming request well. With the increase of the number of connected clients, the average response time and the average delay time of asynchronous requests are not significantly increased, that is to say, the client load increase did not significantly reduce the server's efficiency. Therefore, the model proposed in this paper is feasible.

5 Conclusions

In this paper, the data acquisition in the whole process of a more common function to form a separate implementation of the Server. All the functions which are directly related to the data collection to Client are kept. This structure can improve the reuse rate of software and reduce the code duplication. In the design of the cache management module, we use centralized management of cache files and composite compound application statistics data, so the value of the data collected is improved. In order to describe the dynamic operation of the system better, this paper also defines the internal and external event types that are driven by the drive model. The feasibility of the model is verified in this paper. As mentioned above, the model increases the task of Client and Server communication, but it also brings a lot of functionality and performance that [1] [2] system do not have.

6 Expectation

Server can also be extended, these extensions can not only make the system function more powerful, but also can improve the data collection service. For example, extending function allows user to enable an application to collect data, this makes the data collection to the data security of user minimal. Standardized client data format definition and let more application developers use the client model to collect data can push the compound data of all applications in one device into cloud, and improve the value of the data collected. In addition, you can use local data to provide data services to mobile users, for example, gathering statistics on the use of the applications of the frequency can be used to recommend to remove some of the application is not commonly used, and to clean out free space for mobile phones.

References

[1] ZePeng She. Design and implementation of data acquisition components of mobile application based on Android. Beijing University of Posts and Telecommunications, 2013.

[2] Jie Kang, YiWei Qu. Ajax-based implementation of the client application on the mobile data acquisition Application Platform [J]. Science & Technology for China, 2014(14).

[3] Baidu https://mtj.baidu.com/web/welcome/whitepaper.

[4] Umeng http://dev.umeng.com/analytics/android-doc/integration.

[5] Google https://developer.android.com/reference/android/os/IBinder.html.