Short Message Service for Internet-Mobile Platform

Yingrong Coral Sung
Center of Information and Communication Technology
National Chiao Tung University
Hsinchu, Taiwan
ycsung@cs.nctu.edu.tw

Jui-Feng Sung
Department of Computer Science
National Chiao-Tung University
Hsinchu, Taiwan
rfsung@cs.nctu.edu.tw

***Abstract*—In 2003, the Parlay group proposed a concept and technique of the web service and proposed the other suite of Open Services Architecture (OSA) specifications, Parlay X. The Parlay X Application Programming Interface (API) defines a set of easy-to-use API, which utilizes the web service for application developer to access telecommunications functions more easily. In this paper, we use Parlay X-based Internet-Mobile platform called IBM WebSphere software for Telecom (WsT). To accommodate the telecom services (i.e., Short Message Service (SMS), Multimedia Messaging Service (MMS)) on IBM WsT, a set of API named OpenAPI is implemented. The paper studies the SMS behaviors and investigates the OpenAPI SMS performance. To improve the OpenAPI SMS delivery delay, a timeout timer is implemented. Our research also predicts SMS delivery failure and provides guidelines to select appropriate timeout timer values for OpenAPI SMS.**

Keywords—Open Services Architecture (OSA), Parlay X, Short Message Service (SMS), IBM WebSphere software for Telecom (WsT)

I. INTRODUCTION

In 2003, the Parlay group proposed a concept and technique of the web service and proposed the other suite of Open Services Architecture (OSA) specifications, Parlay X. The Parlay X Application Programming Interface (API) defines a set of easy-to-use API, which utilizes the web service for application developer to access telecommunications functions more easily. A telecom service development platform called IBM WebSphere software for Telecom (WsT) [1] allows telecommunications carriers, equipment providers and application developers to create, deploy and manage voice, video and data services more rapidly and affordably. Based on IBM WsT, we propose a mobile Internet environment for telecom applications and implement a set of APIs based on Parlay X and RESTful web service to support convenient application development named OpenAPI.

Figure 1 illustrates the IBM telecom service execution environment architecture. Based on this architecture, we have developed a WsT application that sends Short Message Service (SMS) messages through OpenAPI. SMS is a narrow-band application, but is an excellent candidate for application-level signaling. Therefore, we consider SMS as the first application in WsT for performance investigation. Specifically, we study the SMS behaviors through OpenAPI, IBM WsT and Chunghwa Telecom SMSC and evaluate the performance of the delay of OpenAPI SMS delivery.

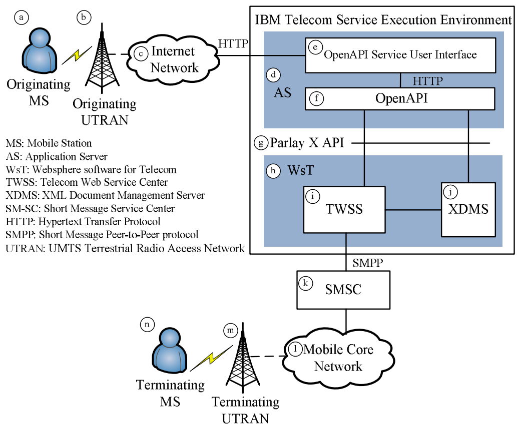


Fig. 1. IBM Telecom Service Execution Environment Architecture . (OpenAPI Short Message Service)

This paper is organized as follows. Section 2 gives an overview on IBM WsT and describes the message flow for SMS. Section 3 presents the performance evaluation of OpenAPI SMS delivery. The OpenAPI SMS failure detection is elaborated in Section 4 and the conclusion is given in Section 5.

II. IBM WEBSPHERE SOFTWARE FOR TELECOM

IBM WsT is a platform providing NGN/IMS standard compliant network services. IBM WsT encompasses the following products: IBM WebSphere Telecom Web Services Server (TWSS), and IBM WebSphere XML Document Management Server (XDMS). In this paper, we use the IBM TWSS to develop WsT applications.

A. IBM Telecom Web Services Server

The TWSS enables service providers to access secure, reliable, and policy driven network services. Therefore, third-party service providers can enhance consumer and enterprise applications (resided in the AS; Figure 1 (d)) through Parlay X. Figure 2 illustrates the TWSS components.

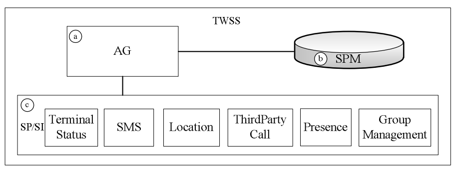


Fig. 2. IBM Telecom Web Services Server Components

The Access Gateway (AG; Figure 2 (a)) provides a common control point for service providers to define, manage, and enforce policies and Service Level Agreements (SLA) for requesters (e.g., third-party services or subscribers).

The Service Policy Manager (SPM; Figure 2 (b)) provides management, storage, and retrieval functions for the policy rules. The SPM can be used to manage definitions of third-party requesters, service definitions, and service relationships. The policy rules enable the service provider to control the behavior of the network or a service at various granularities such as global (all subscribers), group (a subset of subscribers) and individual.

The network services (e.g., Parlay X web service and Usage Record web service) are located in Service Implementations (SI; Figure 2 (c)).

The Service Platform components (SP; Figure 2 (c)) provide common service implementation functions such as admission control, traffic shaping, fault alarm, network resources and notification management. These common, reusable functions are shared within the application server by all deployed SI.

B. *Message Flow for SMS through OpenAPI*

Figure 3 illustrates the SMS message flow. The originating MS (Figure 3 (A)) sends the HTTP SMS request to OpenAPI through OpenAPI SMS User Interface (OpenAPI SMS UI; Figure 3 (C)). The SMSC (Figure 3 (I)) receives SP/SI (Service Platform/ Service Implementations) request and sends the message through Mobile Core Network to terminating MS (Figure 3 (J)). The originating MS communicates with the Application Server (AS; Figure 3 (B)) via the Hypertext Transfer Protocol (*HTTP*). The AS communicates with the IBM WsT (Figure 3 (E)) via the Simple Object Access Protocol (SOAP). And the IBM WsT communicates with the SMSC (Figure 3 (H)) via the SMPP.

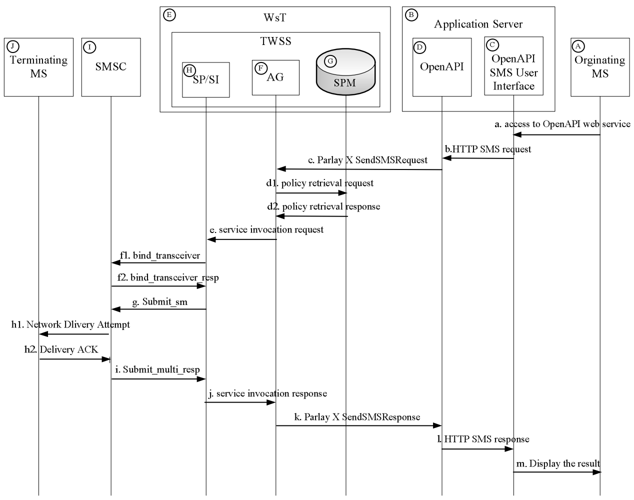


Fig. 3. OpenAPI SMS Message Flow

C. *Mobile Terminated SMS Call Flow*

Figure 4 illustrates the mobile terminated SMS call flow [2,3]. The details of message flow are given below.

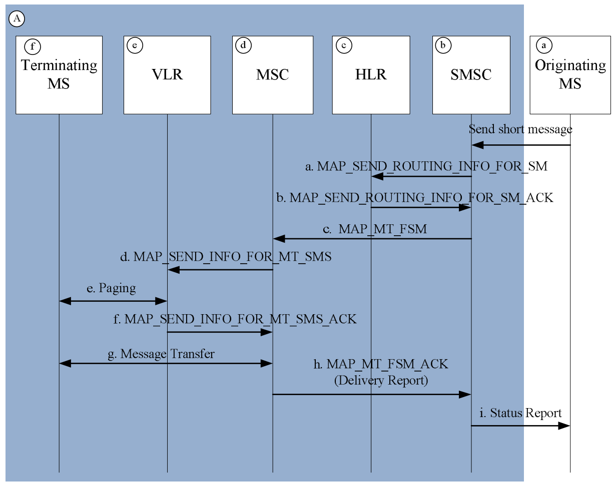


Fig. 4. Mobile Terminated SMS Call Flow

III. OpenAPI SMS Delivery Delay

In this section, we evaluate the performance of OpenAPI SMS delivery delay that is the delay form originating MS to the terminating MS. In the SMS delivery, if a short message transmission fails (i.e., Step h1 in Figure 3), the SMSC retransmits the short message after a waiting period. The SMS retransmission may repeat several times until the short message is successfully delivered. In Chunghwa Telecom, the waiting period is set to four minutes and the maximum number of retransmission is 360 times.

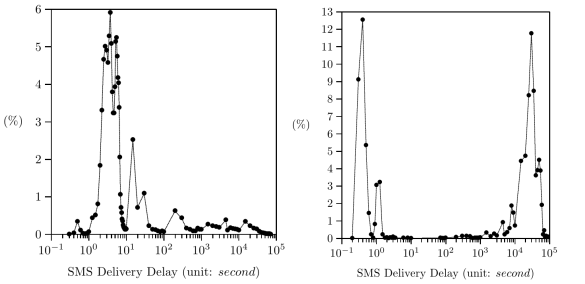
TABLE I. SMS Transmission Delay of Delivery Status Code

Delivery Status Code	Number of SMS	Percentage (%)
0 (Delivered success)	70455	94.03%
11 (Transmission timeout)	1	0.001%
12 (System failure)	796	1.06%
13 (Unknown subscriber)	1265	1.69%
14 (Absent subscriber)	1707	2.28%
15 (Memory capacity exceeded)	60	0.08%
71 (Call barred)	35	0.047%
72 (Send SMS to international SMSC)	321	0.43%
74 (Facility not supported)	99	0.13%
75 (Protocol error)	2	0.003%
77 (MS busy for MT SMS)	1	0.001%
81 (Other failure)	181	0.24%

We measure the OpenAPI SMS delivery delay though OpenAPI SMS application server, IBM WsT and Chunghwa Telecom mobile network on 2012 July 30, summarizing the 74923 short messages statistics into TABLE I. The IBM WsT recognizes the delivery status codes 0 (delivered success) and 72 (send SMS to international SMSC) as delivery success and other delivery status codes (i.e., delivery status code 11, 12, 13, 14, 15, 71, 74, 75, 77, 80 and 81) as delivery failure. As shown in Table I, 70776 short messages are successfully delivered and 4147 short messages are delivered fail.

Figure 5 (a) and Figure 5 (b) illustrate the SMS delivery histograms for the success cases and failure cases respectively. Among the 70776 successfully delivered short messages, Figure 5 (a) shows that 94.58% of short messages (i.e., 66943 short messages) delivered successfully without SMS retransmission in $10^{-1} – 10^{2}$ seconds (4.94 seconds in average). The mean SMS delivery delay of success cases is 5.14 minutes.

Figure 5 (b) shows that the mean SMS delivery delay of failure case is 5.58 hours. Most failure short messages delivery are delivered in $10^{-1} – 10^{1}$ seconds or $10^{3} – 10^{5}$ seconds.



(a) Delivery Success Histogram (b) Delivery Failure Histogram

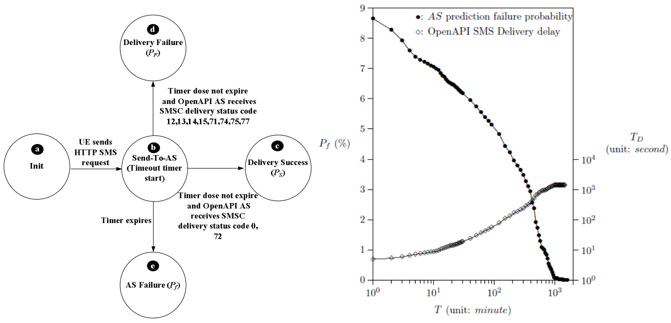
Fig. 5. SMS Delivery Histogram

IV. OpenAPI SMS Failure Prediction

In OpenAPI SMS delivery, if a short message deliver fails, the SMSC retransmits the short message until the short message is successfully delivered. If the short message keeps delivering fail, the originating MS will keep waiting until receives the response (i.e., Step m. in Figure 3) from the OpenAPI application server. As the measured results presented in section III, the mean of OpenAPI SMS delivery delay with SMS retransmission mechanism costs 5.58 hours. Thus, it may take long time for originating MS to detect the delivery failure. To solve this issue, we specify a timeout timer in OpenAPI application server to predict the SMS delivery status. If the timer value is set too large, it takes long time before the originating MS detects the SMS delivery failure. If the timer value is set too small, it may lead to misjudgment because the timer expires before the originating MS receives the response from the OpenAPI application server. In this chapter, we analyze the timeout timer and select an appropriate timer value to improve the performance of OpenAPI SMS delivery such that the prediction inaccuracy rate is limited within an acceptable range and keep originating MS not waiting so long.

Based on the OpenAPI SMS message flow shown in Figure 3, Figure 6 (a) illustrates the SMS delivery state transition diagram which is used in OpenAPI application server to indicate the status of every short message delivery.

Figure 6 (b) illustrates the effect of T on P_f and T_D. It is indicated that the AS-Failure prediction failure probability P_f approaches to zero if the timeout period T is larger than 24 hours. Furthermore, the SMS delivery delay T_D increases as timeout period T increases. It is suggested to set T to 13 minutes. In the suggested case, the OpenAPI AS predicts the SMS delivery status in 10.16 seconds in average and the inaccuracy rate of delivery status prediction is decreased to 1.31%.



(a) SMS Delivery State Transition Diagram (b) Effect T on P_f and T_D

Fig. 6. SMS Delivery State Transition Diagram and Effect T on P_f and T_D

V. Conclusion

This paper presented how to integrate the IBM WebSphere software for Telecom (WsT) with the mobile telecom network and internet. To access the Parlay X web services offered by the WsT, we implemented a set of API based on Parlay X API named OpenAPI. OpenAPI is an interface between mobile application and IBM WsT which allows application developers to create new services or applications without knowing the details of the telecommunication network. We use the Parlay X SMS as an example to illustrate how a new mobile application can be easily created in the IBM WsT platform.

In web-based SMS, the application receives the SMS delivery status response from SMSC. We studied the SMS behaviors through OpenAPI, IBM WsT and Chunghwa Telecom SMSC and evaluated the performance of the OpenAPI SMS delivery delay. Our study provides guidelines to select appropriate timeout timer values to improve the delay performance of OpenAPI SMS delivery. It is suggested to set timeout timer value to 13 minutes which predicts the performance of OpenAPI SMS delivery delay in 10.16 seconds in average and the inaccuracy rate of delivery status prediction is decreased to 1.31%.

Acknowledgment

This work was sponsored in part by IBM Taiwan, Chunghwa Telecom. We thank Y.-B. Lin for his guidance and discussions on this work.

References

[1] IBM WebSphere software for Telecom, Second Edition. http://publib.boulder.ibm.com/infocenter/wtelecom/v7r0m0/index.jsp

[2] 3GPP. Technical realization of the Short Message Service (SMS). TS 23.040, 3rd generation Partnership Project (3GPP), September. 2008.

[3] ETSI TS 123 040 V7.2.0, "Digital cellular telecommunications system (Phase2+); Universal Mobile Telecommunications System (UMTS); Technical realization of Short Message Service (SMS)" (3GPP TS 23.040 version 7.2.0, Release 7), March, 2009