A Web-Based Application to Verify Open Mobile Alliance Device Management Specifications

Jose F. Mejia Bernal
Department of Automation and Information
Politecnico di Torino
Turin, Italy
jose.mejiabernal@polito.it

Paolo Falcarin
Department of Automation and Information
Politecnico di Torino
Turin, Italy
paolo.falcarin@polito.it

Maurizio Morisio
Department of Automation and Information
Politecnico di Torino
Turin, Italy
maurizio.morisio@polito.it

Abstract—**In this paper, we describe the implementation of a web-based application supported by SyncML protocol, which is able to execute management operations in different kind of devices. This application is characterized by a set of management options, in order to test and verify whether specific mobile devices implement Open Mobile Alliance(OMA) specifications. The elements that compose our web-based application, have been designed taking into account requirements derived from different kind of operations executed in mobile devices. We have also defined a set of tests, that provide a suitable way to establish whether a specific device supports device management operations according to OMA specifications. The lack of both available code and reliable applications for device management was an important motivation for our work.**

Keywords-component; SyncML; Management Operations; Device Management

I. INTRODUCTION

The increasing proliferation of mobile devices and the intention of defining universal standards related to mobile market, motivate many companies to support the Open Mobile Alliance, with the purpose of stimulating the fast and wide adoption of a variety of specifications, some of them related to management devices. The implementation of OMA specifications [1] results in benefits to the end-users, providing easy interoperability across operators and mobile terminals.

Many applications based on different kind of technologies, also require new solutions for dynamically extending management device functionalities. Such functionalities can be visible by implementing web-based features, characterized by supporting management operations in mobile devices and a reliable synchronization protocol, adapted to the management requirements.

In order to extend management device functionalities, we have seen that SyncML [2] is a synchronization protocol able to provide a suitable communication channel between mobile devices and web functionalities based on management operations. A management server for mobile devices, defined as server DM, includes the main characteristics of such protocol. Such server implements every essential component to manage mobile devices through web-based applications.

The main management operations that we have identified, were defined within a functionality called Test Case Manager. Such functionality includes management operations that can be executed on specific mobile devices identified by their International Mobile Equipment Identity (IMEI). The set of tests associated with this functionality are:

- Base Operations: Executes management operations on single nodes that define the logic device structure.
- User Interaction: Sends information that will be displayed on the device screen, in order to allow the user answers according to the options provided.
- Management Properties: Retrieves device properties defined by the SyncML protocol.
- Tree Discovery: Explores the whole logic device structure defined as a management tree.
- Notification Message: Sends a notification displayed on the device screen.

The rest of this paper is organized as follows. Section II includes related work. Section III describes the main components of the SyncML protocol. Section IV describes characteristics of the architecture and the components of the server DM. Section V describes the architecture of our application. Section VI specifies the main tests characteristics. Finally, Section VII includes conclusions and future work.

II. RELATED WORK

In [3], the authors present an architecture for the management of mobile and non-mobile devices in an enterprise. In such an architecture, they integrate SyncML with SNMP to propose a solution that allows maintaining existing investment in the enterprise management system, while providing multi-device coordination and management transaction control.

According to [4], existing mobile e-mail solutions have some deficiencies, such as poor mobility. To avoid these problems, they propose a way to design and implement an Enterprise Mobile E-mail System with SyncML protocol. Their approach shows that SyncML characteristics are able to provide a reliable way to implement good solutions for mobile data applications supported by a robust synchronization protocol.

Device management testing is an important matter for providing acceptable services to end users. In [5], authors present an open-source agent tool based on SyncML model and evaluate the performance and cost of this approach in the context of limited devices. Such approach introduces a device management more related to the management of new services deployed in devices with limited resources. On the other hand, our web-based application is also based on device management, but in our case we provide a way to verify whether specific mobile devices are able to support device management operations defined by OMA specifications.

In [6] the authors designed and developed a data synchronization system based on the SyncML protocol. They also evaluate the throughput of the system using Stochastic Petri Nets, to analyze the relationship between the arrival rate and the system resources. In this way, they are able to evaluate various performance measures in different situations.

Our work is different from the above mentioned approaches, because we focused on an extensible web-based approach to test and verify OMA specifications compliance. Literature about approaches related to our web-based approach is limited, mainly due to many contributions are carried out by private companies, in this way important contributions remain unpublished.

The lack of both available code and robust web-based approaches able to verify OMA device management specifications, was an important motivation to make an interesting contribution. Such device management specifications are related to the way in which a mobile device should behave according to specific management operations.

III. SyncML Protocol

SyncML [7] is an industrial initiative, based on Internet and Web technologies, in order to develop and promote a data synchronization protocol for all types of devices, like PDAs, PCs and mobile phones. SyncML specification includes two primary objectives:

- Network data synchronization with any kind of mobile device.
- Synchronization of mobile devices with all data on the network.

To achieve such objectives, SyncML was designed as one platform, one net and one protocol for different kind of applications, to allow synchronization any-to-any and access to many types of data. SyncML is based on XML and characterized by:

- Working on any network used from mobile devices.
- Including different type of transport protocols.
- Synchronization with the multiple data warehouses.
- Independence of the programming language.
- The exchanged data are represented in a binary format in order to reduce memory requirements.

A. *Protocol Characteristics*

Mobile operators have to provide an easy and fast device configuration, also called device management. Device management is a generic concept for systems related to configuration, control, update and access to different settings of mobile devices.

SyncML defines a protocol for different synchronization procedures, represented as a messages sequence. A SyncML message is a well formed XML document. Such document is composed of heading and body. The heading specifies the path and the version information and the body is a container of one or more SyncML commands.

B. *Message Sequence*

The message sequence in Figure 1, is composed of Alert phase (Management initialization not requested), Set up phase (Authentication and information exchange) and Management phase (Management device) [8].

- Transaction 1 (Alert Phase): SyncML is based on not requested alerts through a mechanism called notification initiation alert [9]. Such mechanism permits the management server to start a management session.
- Transaction 2 (Set-Up): Involves both the client request and the server answer. The initial client request [10] contains device information, client credentials(Authentication) and session Alert.
- Transaction 3 (Set-Up): The server answers the initial client request by sending its credentials, also can transmit interaction commands with the answer and initial management data.
- Transaction 4 (Management): It is requested only whether the management data or the user interaction commands are sent in the previous message. The client will include the results of the management message sent in the previous transaction.
- Transaction 5 (Management): Requested whether the transaction 4 was already initiated. This transaction is executed when the management session is closed or a new management iteration starts.

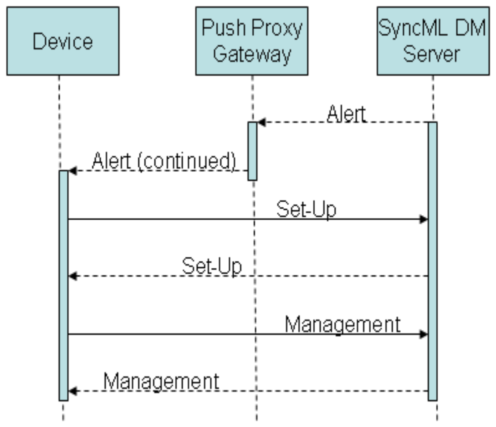

Figure 1. Transactions

C. *Device Management*

The SyncML protocol is able to identify different kind of messages and their contents (management operations). A data structure, known as Management Tree [11], was defined in the device to facilitate the identification and execution of management operations in the tree nodes.

The tree data structure permits to address SyncML messages. Device management tree is a data structure of static or dynamic objects. Static objects are included in the device fabrication and can not be cancelled, instead the dynamic objects, can be added or cancelled in any moment.

Each object of the management tree has a set of properties that define the metadata information object. Such objects can be manipulated through valid SyncML messages using Add, Get, Replace, Delete or Copy commands [12].

IV. Device Management Server

The management server implements a reliable structure to develop applications based on mobile device management. The server integrates elements that involve structural and functional aspects based on synchronization. It was designed with the purpose of supporting all the functionalities defined in the SyncML protocol.

This is an important section, because it shows how is the interaction between: the mobile device, the web-based application and the server DM. The Device Management Server is an important component to establish a reliable communication channel between our web-based application and the mobile device.

A. *Architecture*

The server architecture [13], is typically represented by the next set of components: Transport Layer (HTTP), Protocol Layer (SyncML), Server Layer, Application layer (interaction rules between the server DM and third-party applications) and Framework (implements and provides services).

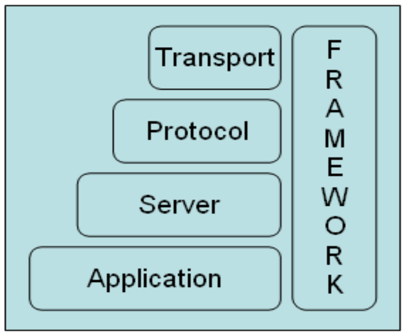

Figure 2. Server DM Architecture

B. *Management Characteristics*

1) *Request Execution:* Device management sessions can be initiated from the device or being requested from the server [14]. When the server initiates a management session, the execution flow involves the next steps (Fig. 3):

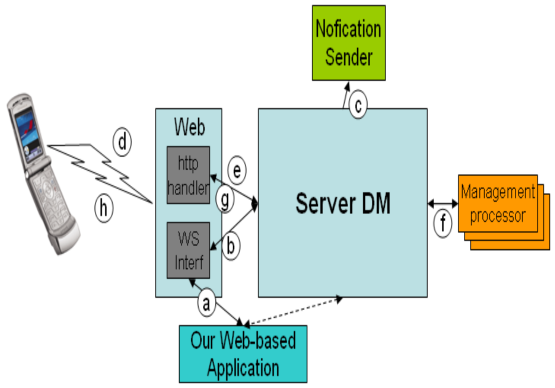

Figure 3. Execution flow of OMA DM request

a) The application starts a new management operation on some specific device.

b) The web service invokes a specific interface, in order to deliver the call to the Server DM.

c) The Server DM builds the notification message and sends it to the mobile device.

d) The device receives the notification message and starts a new management session.

e) The HTTP handler is ready for both opening the management session in the server and starting the message composition.

f) In order to send management commands to the client, the Server DM selects and calls the management processor more adapted to the kind of operation.

g) The response message is translated in a device management message.

h) The response message is sent to the device.

The management process starts again in the step *d)* with a new client message; so the commands and the results exchanged between the client and server belong to the same session up to the server does not send more commands.

2) *Message Processing:* Some transformations are carried out in the message when it arrives to the server. In the XML format, the message is still difficult to manage, therefore it is transformed in a tree object that represents the message. After such transformation, the message arrives to the server. In a similar way, an answer message is delivered from the server and further translated to XML format.

3) *Implementation of Management Operations:* A management operation is a sequence of commands that the server transmits to the device in order to execute some operation [14]. The server manages sessions and operations related to the device. When a client starts a new management session, the server engine selects the processor through the responsible selector. Such selector will make a decision based on the contents of the initial device

information transmitted from the client. We have suggested two types of implementations based on the previous aspect:

a) Allocation of the processor according to the device identification. Whether the identification is not possible, a default processor will be assigned.

b) Association of the processor depending on the management operation.

When a management operation is executed in the client, the results state and maybe the data are sent back.

V. Implementation

We have proposed the architecture in Figure 4, in order to implement a web-based application supported by SyncML protocol, able to execute management operations in different kind of devices. Our web-based application is characterized by a set of management options, to verify whether specific mobile devices implement OMA device management specifications.

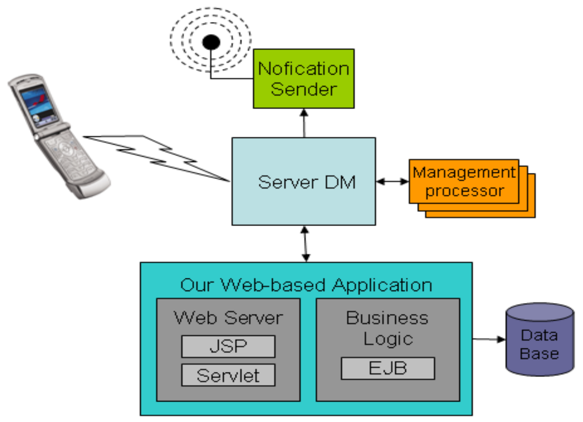

Figure 4. Architectural overview

The implementation phase considers the elements that share our web-based application functionalities. Such functionalities regard every component involved in the interaction between management procedures.

Management operations are executed through web components that provide a specific processor for each type of operation. Each component provides an interface implementation in charge of: opening and closing the session related to each operation, receiving the selected operations and sending back the results.

A. Web-Based Components Developed

Web-Based components are in charge of carrying out business logic activities, depending on management operations. Such components have been developed according to the tests characteristics described in the next section. The development of this part is characterized by:

- Java-based components [15] that consider in which way our web-based application should react and process management operations results, depending on the management processor involved in the operation. Such elements provide support to reception, elaboration and delivery of the information, according to the interaction between the user and our application.
- Dynamic web-component that validates the testing management procedure. Such web-component is in charge of both, identifying the type of test executed from the user and checking the adapted procedure to carry out the operation.

The management operation activity begins when specific parameters are received from the user-interface. Such parameters store different type of data, which are received to apply the respective validation procedure. The management activity is divided in the next steps:

1) Once the validation web-component has received every parameter requested, it looks for identifying the type of operation that the user wants to execute, in order to associate the current operation with the adapted processor.

2) When the management operations are not related to the Tree Discovery Processor, the validation web-component invokes a procedure known as startSmsManagementSession. Such procedure uses SyncML features, implemented in the device management server, in order to initiate the management session.

Considering the way in which a management operation starts, according to SyncML specifications, the purpose of the startSmsManagementSession invocation is creating a new management session, identifying the device and sending a message to initiate the session. The message delivery is executed through a specific method that belongs to the device management server.

3) A new session creation involves the identification of the type of processor defined from the validation component. Once identified the processor, the management session is ready to start by invoking the respective procedure. According to SyncML specifications, our application must add an alert before any operation. Such alert will be a signal by which the mobile device will know that our web-based application wants to initiate a management session.

4) As we mentioned before, there is a specific processor associated with each kind of operation. In this way, the device management server handles this matter through a XML configuration file. Such file is in charge of creating an instance of the class that defines the processor for each case.

Once the processor related to the operation in execution, has been identified, it will be able to receive either a list of operations or a single operation.

Such processor continues the management activity and sends the operations to the device, after that, the device has to send back the respective results to our web-based application. The results are saved in a XML file, in order to be further displayed through the user-interface component.

5) When the processor terminates the management session, because there are not more operations to be executed, or results to be received, a web-based component will be able to both interpret the information contained in the XML file and send it back in a format adapted to be displayed.

At this point, the web-based component is able to save the management session in the database [16]. By saving each session, we have the possibility of recovering data and reloading previous sessions. Such data recovery is useful to execute further analysis related to the way in which some specific device supports OMA specifications.

B. Management Components Developed

Device management server is composed of elements that implement management and configuration characteristics, so they react depending on the management activities. According to the device management server specifications, such components are known as processors. In order to offer both proper environment and adapted functional conditions to the server, we have designed two kind of components:

1) The first component involves a set of classes that defines the basic structure for each type of processor, according to the operation in execution. For each management operation, we have defined one processor with specific behavior. Since the processor scope is related to handle management operations, we implemented its main functionalities through the next methods:

a) getNextOperations: Handles everything related to sending management operations to the device management server.

b) setOperationsResults: Provides the management operations results.

In order to retrieve management operations results for further analysis, it has been necessary developing some components capable to store such results in a xml file. Such functionality has been added with the purpose of facilitate both portability and management information.

2) The second component considers XML configuration files, which have been created according to tests characteristics. Each test has been associated with a specific configuration file. Since each test is identified through a specific name, the server will be in charge of associating one configuration file with the selected test. The XML configuration file is an element that associates the processor more adapted with the test that our web-based application has to carry out.

VI. Tests Characteristics

This section covers a detailed description of the main tests characteristics proposed in our web-based approach:

A. Base Operations

To implement such operations, we developed a set of processors associated with each option. We have considered the next kind of operations:

1) Get Device Detail: Sends to the server a list of operations to extract the nodes details.

2) Get Node: Extracts one existent node. The server has to confirm the operation by sending back a state code identified by the number 200.

3) Get inexistent node: Executes the operation related to inexistent node extraction. The server has to send back a state code identified by the number 404(not found node).

4) Add Node: Adds a node in the management tree that represents the logic structure of the device.

5) Delete Leaf Node: Deletes a leaf node like ./DevInfo/DevId from the management tree. In this case, the server sends back operation results.

6) Delete Generic Node: Deletes some root node.

7) Delete Inexistent Node: Executes the operation related to delete some inexistent node.

8) Replace Leaf Node: It is similar to operation in 5), but in this case the server looks for replacing a leaf node.

9) Replace Generic Node: It is similar to operation 6), but in this case the server looks for replacing a generic node.

10) Replace Inexistent Node: Involves inexistent node replacement.

B. User Interaction

The SyncML protocol specifies the next kind of interactions with the user, according to the notification and reception of user confirmation:

- User notification is associated with some action.
- User confirmation for executing some operation.
- Informs the user about the next operation.
- Enables the user to select one or more items.
- Displays the notification progress related to some specific action.

Such notifications are translated in alerts that can be sent only from the server side. When some interaction with the user is executed, the server is informed of the interaction result through a state message.

The possible interaction states are: (214) Interaction cancellation from the user, (408) Answer time expired, (406) User interaction is not supported and (416) device limitation. According to the management process and expected results, the next set of management operations executed on the device have been defined:

1) User Interaction: An alert message that contains two options should be displayed on the screen.

2) User Input Text: The device displays a text field.

3) User Input Choice: The device displays a set of options, one has to be selected from the user.

4) User input date: The device displays a text field to be filled with the date.

C. Management Objects Properties

The management objects properties are used to provide meta-information related to the object involved in the management process.

TABLE I. MANAGEMENT OBJECTS PROPERTIES

Property	Description	Type
ACL	Access Control List	Mandatory
Format	Specifies how the object values should be interpreted	Mandatory
Name	Object name in the tree	Mandatory
Size	Object in bytes	Mandatory for leaf objects / Not mandatory for node objects
Tittle	Tittle	Optional
TStamp	Time stamp, date and time of the last change	Optional
Type	MIME type	Mandatory for leaf objects / Optional for node objects
VerNo	Version number	Optional

D. Tree Discovery Operations

Management servers can explore the tree structure by using the GET command. If the achieved object has leaf objects connected, then their names will be provide as a consequence of the GET command. If there are not leaf objects, the node object has to provide a value.

E. Notification Message

Many devices can receive not requested messages, sometimes called notifications. A management server can use such notification functionality in order to stimulate a new connection with the client. In other words, notification message is a way to allow the server sends an alert to the client, with the purpose of starting the synchronization without the execution of complex procedures.

VII. CONCLUSIONS AND FUTURE WORK

SyncML is a powerful protocol, since the interoperability between devices, transport protocols and network databases permits the implementation of important characteristics for the applications development, that integrate various management components related to mobile devices.

Our web-based approach integrates suitable and robust components to test and verify whether mobile devices implement Open Mobile Alliance (OMA) specifications. We have proposed a powerful approach that provides a reliable way to establish whether a specific device supports device management operations according to OMA specifications.

A robust web-based approach is a perfect way to facilitate the execution of management operations in specific mobile devices. Such kind of approach involves a suitable and reliable way to coordinate both: the execution of the proposed set of tests, in some specific mobile device, and the management of the results retrieved from the selected device.

As future work, we are looking for integrating a rule based engine with our web-based architecture. In this way, our web-based approach will be able to make automatically management decisions, according to a set of pre-established rules based on management tests results.

REFERENCES

[1] Open Mobile Alliance (2007), retrieve March 16, 2009, from http://www.openmobilealliance.org

[2] U. Hansmann, R. Mettala, A. Purakayastha, P. Thompson, "SYNCML: Synchronizing and Managing Your Mobile Data". Prentice Hall, Sept 2002.

[3] S. Adwankar, S. Mohan, V. Vasudevan, "Universal Manager: Seamless Management of Enterprise Mobile and Non-mobile Devices", IEEE International Conference on Mobile Data Management (MDM'04), pp. 320-331, 2004.

[4] Jing Xie, Dan Yu, Shilong Ma, "A Novel Enterprise Mobile E-Mail System Using the SyncML Protocol", In seventh IEEE/ACIS International Conference on Computer and Information Science, pp. 391-396, 2008.

[5] R. Stute, 0. Festor. B. Zores, "An Extensible Agent Toolkit for Device Management", In proceedings of the IEEE/IFIP Network Operations and Management Symposium, April 2004.

[6] Byung-Yun Lee, Gil-Haeng Lee, Young-Sun Kim, "Modeling and Analysis of SyncML Server System Using Stochastic Petri Nets", In proceedings of the Third International Conference on Wireless and Mobile Communications (ICWMC'07), pp. 60-66, March 2007.

[7] "SyncML Representation Protocol vl..1" (2002), retrieve March 16, 2009, from: http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_represent_v11_20020215.pdf

[8] "Device Management Protocol vl..1", retrieve March 16, 2009, from http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_dm_protocol_v11_20020215.pdf, February 2002.

[9] "Device Management Notification Initiated Session", retrieve March 16, 2009, from http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_dm_notification_v11_20020215.pdf, February 2002.

[10] "Device Management Standardized Objects", retrieve March 16, 2009, from http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_dm_std_obj_v11_20020215.pdf, February 2002.

[11] "Device Management Tree and Description", retrieve March 16, 2009, from http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_dm_tnd_v11_20020215.pdf, February 2002.

[12] "Device Management Representation Protocol",retrieve March 16, 2009, from http://www.openmobilealliance.org/tech/affiliates/syncml/syncml_dm_represent_v11_20020215.pdf, February 2002.

[13] "DM Server DemoUser", retrieve March 16, 2009, from http://download.forge.objectweb.org/sync4j/DM_Server_DemoUser.pdf, July 2006.

[14] "DM Server developer guide", retrieve March 16, 2009, from http://download.forge.objectweb.org/sync4j/funambol_dm_server_developer_guide.pdf, September 2006.

[15] Web Site Sun Microsystems, retrieve March 16, 2009, from http://java.sun.com/

[16] Web Site MySQL, retrieve March 16, 2009, from http://www.mysql.com/