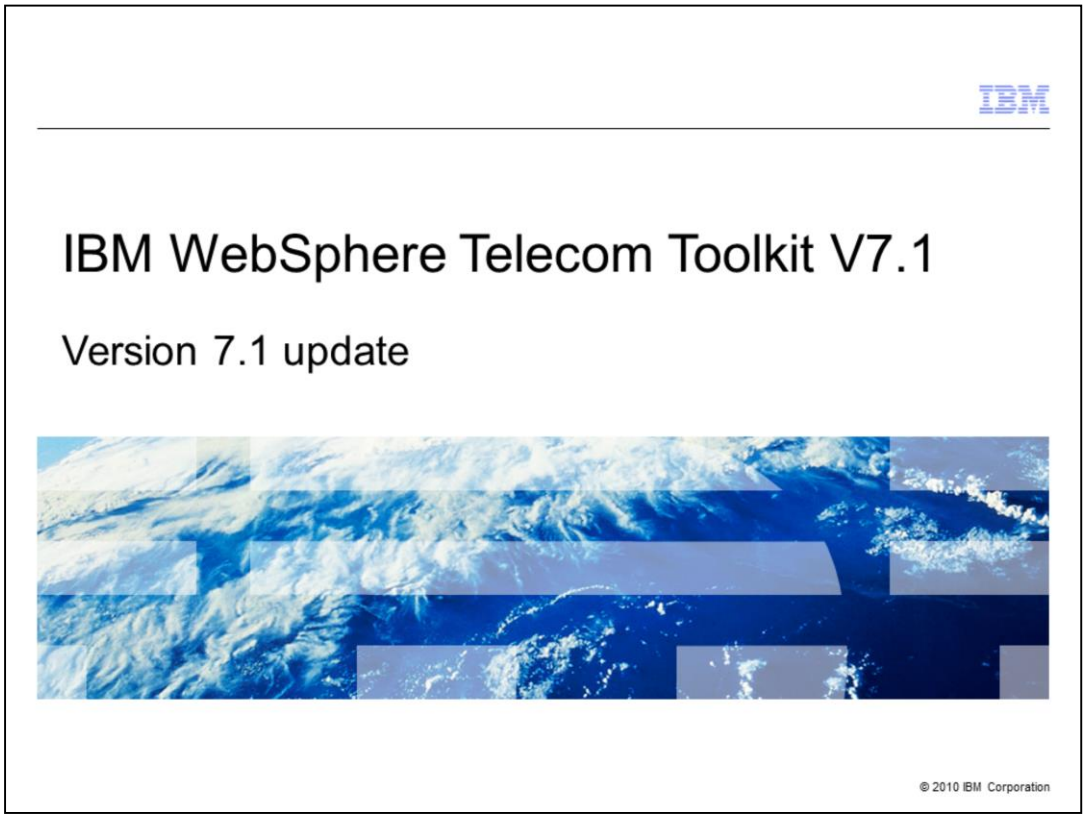
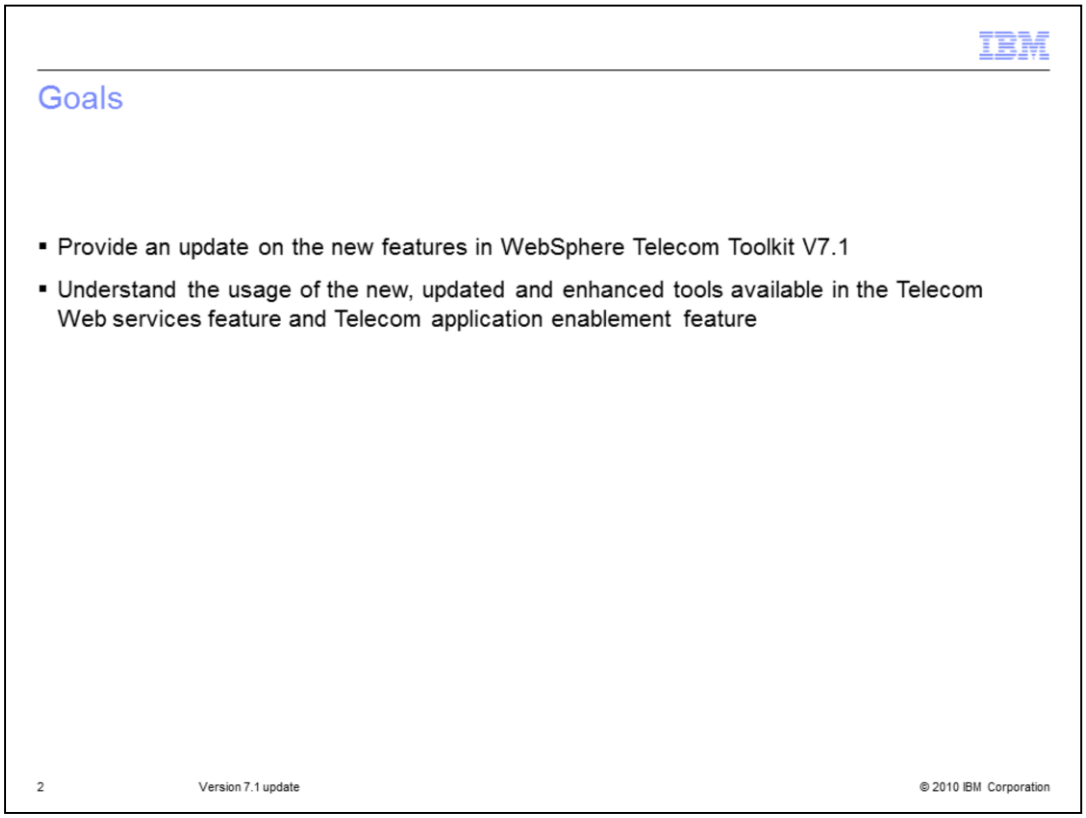
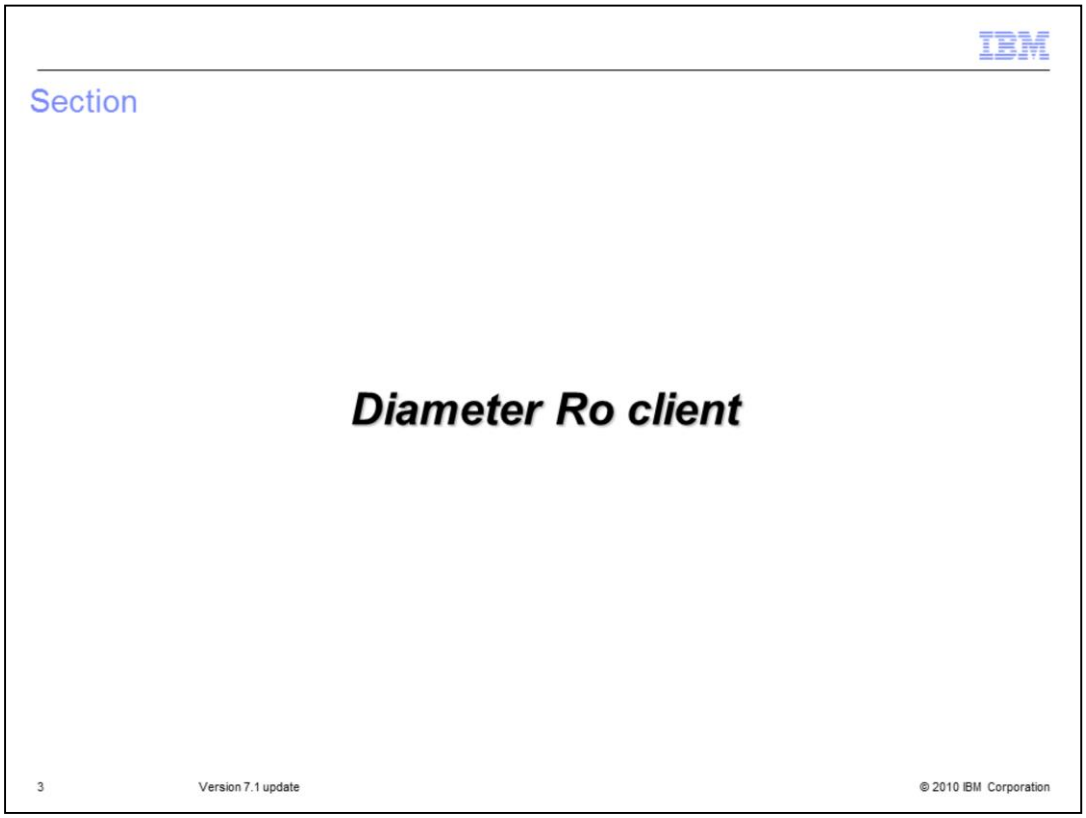


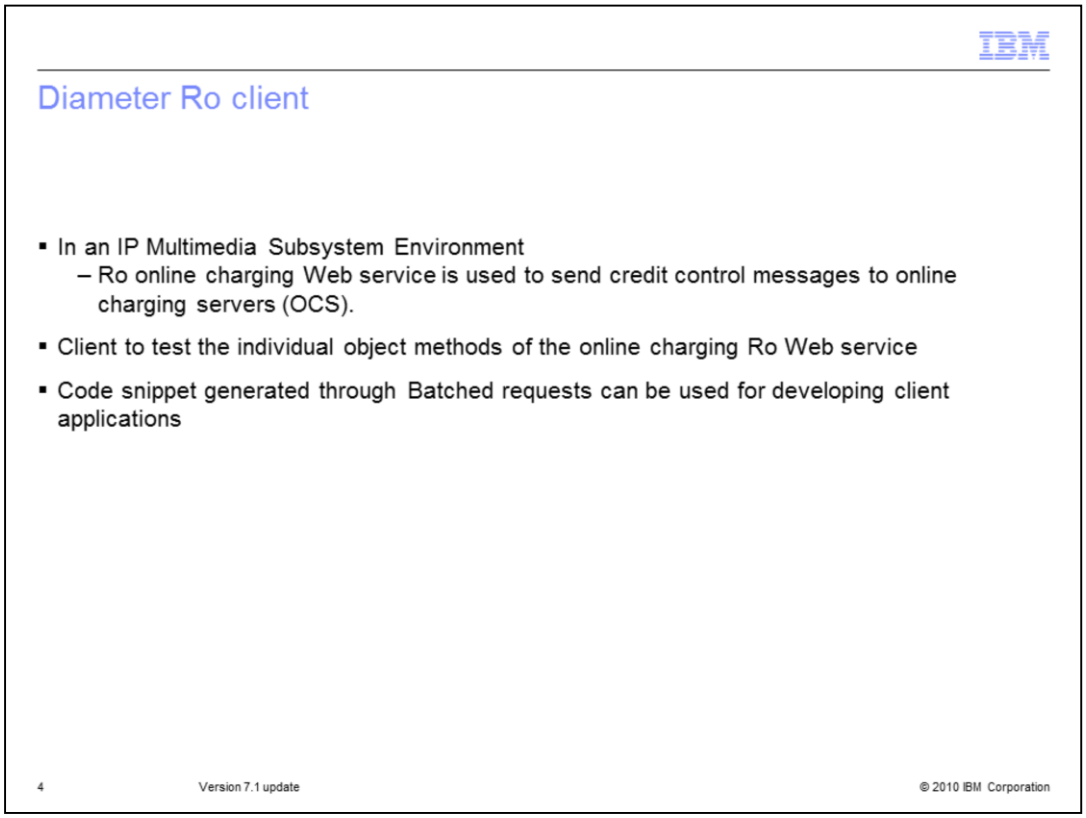
This presentation will provide an overview of the new features added to the IBM WebSphere® Telecom Toolkit version 7.1.



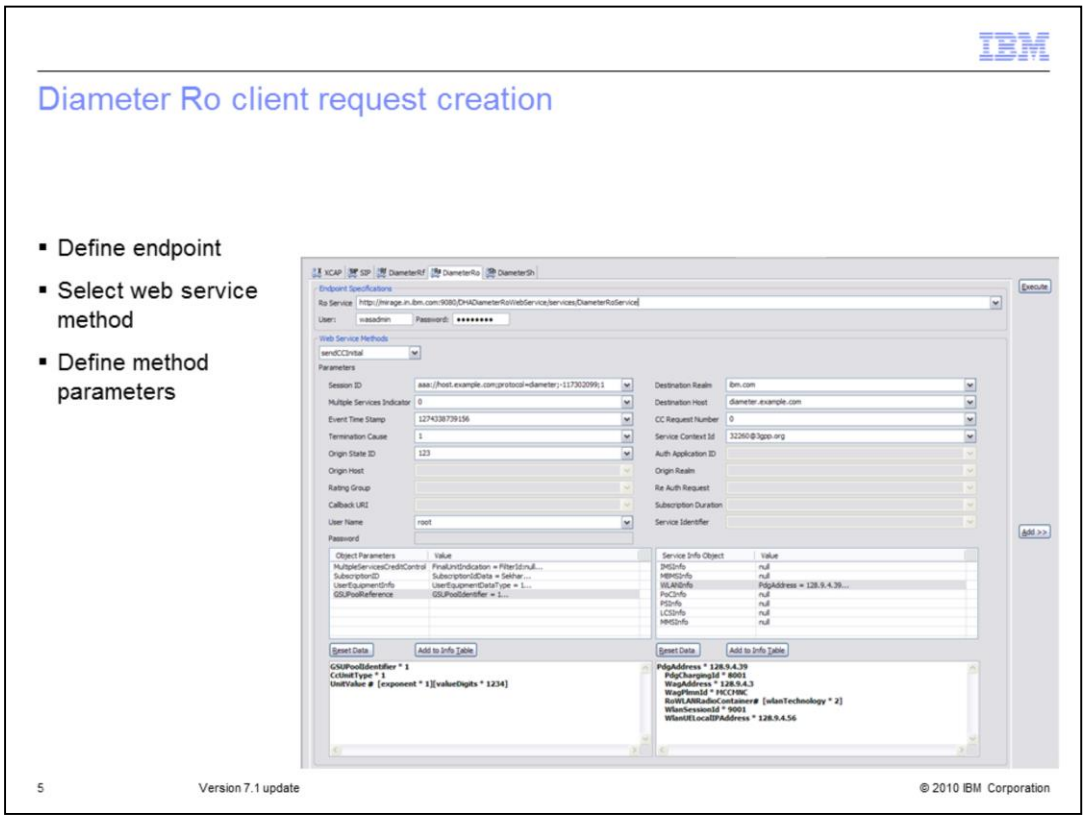
The goal of this presentation is to introduce the new features in the telecom Web services feature and telecom application enablement feature of the WebSphere Telecom Toolkit version 7.1. These new exciting features simplify various aspects of developing telecom client applications using telecom Web services and aid in the testing of applications by use of an integrated simulator. The simulator simulates a real telecommunication network for functional testing.



The agenda for this presentation will start with an overview on Diameter Ro client which is added to the existing Batched Script editor in Telecom application enablement feature followed by an overview of the new REST style sample added to the Telecom Web services feature.



The Diameter Ro client tool provides a user interface driven way to test the individual object methods of the Web services description language (WSDLs) of the online charging Ro Web service. In an IP Multimedia Subsystem environment the Ro Web service is used to send credit control messages to online charging servers (OCS). Just like the other client tools, the Batch request tool provides a feature to copy the code snippet from the test that can be reused in developing client applications. A more detailed description about the Batch requests tool is provided in Client tools presentation under 7.0 version.



The figure in the slide shows the screen capture of Diameter Ro online charging request section and the various user interface controls available to configure Ro requests. Creating a Ro Web service request involves three steps that include:

- Defining the Endpoint Specification for the Ro service which is the URL of the Diameter server where the Ro service endpoint is configured. The User and Password information are used for authenticating the request.

- Selecting a method from the Web service Methods drop down.

- Defining all the parameters of the method. The common parameters for all the Ro Web service methods include: -

Session ID – Is a globally unique identifier that identifies a user session. For example aaa://host.example.com;protocol=diameter;-117302099;1

Destination Realm – Is the realm the subscriber belongs to. The destination realm is a required parameter and must be a fully qualified domain name.

Multiple Services Indicator – Indicates whether the Diameter credit-control client is capable of handling multiple services independently within a session or sub-session

Destination Host – Is the fully qualified domain name of the destination host. This value is generally not specified when using realm routing. This input parameter is optional. *Examples*: sipintel15.city.example.com, diameter.example.com

Event Time Stamp – Is the timestamp to record the time the event occurred.

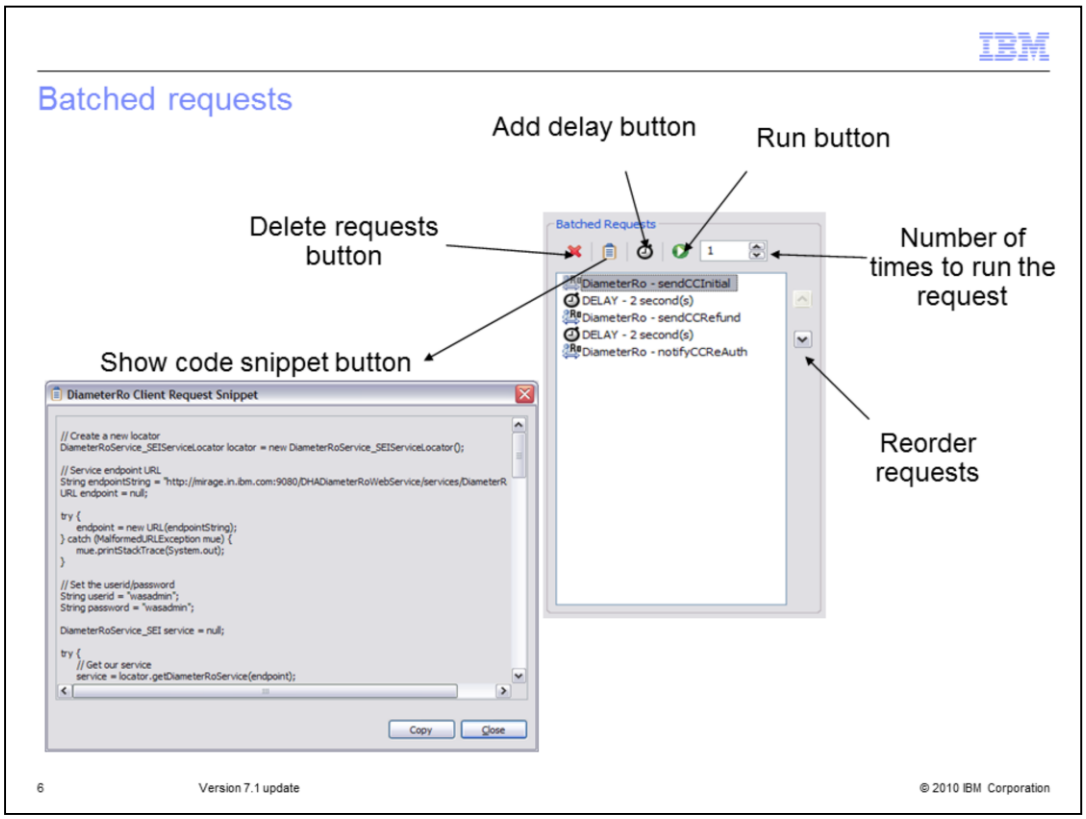
CC Request Number – Is the numbered request within a session. Set this value to 0 for request types of Initial and Event. Increment this value by one for each subsequent request within a session.

Termination Cause – Indicates the reason why the session was terminated.

Service Context Id – Is the specific request document that this request follows. For IMS charging: “32260@3gpp.org”

Origin State ID – holds the value to track the incremented value of possible times the client has lost the state or a possible reboot has occurred

User Name – Is the private user identity if available in the node



The Batched Request section of the client editor is common to all client test tools. Multiple client requests from different clients can be run together through the Batched Requests section. The Batched Requests section provides some additional useful features that are explained through the figure in the slide. These additional features include:

- Performing validation on request parameters in the request before adding to the request batch

- Reorder Requests button for reordering requests already added to the batch

- Number of Execution Iterations field to increase or decrease the number of times the requests need to be run

- Execute button to run the batched requests

- Add delay button to add time delay in seconds between requests

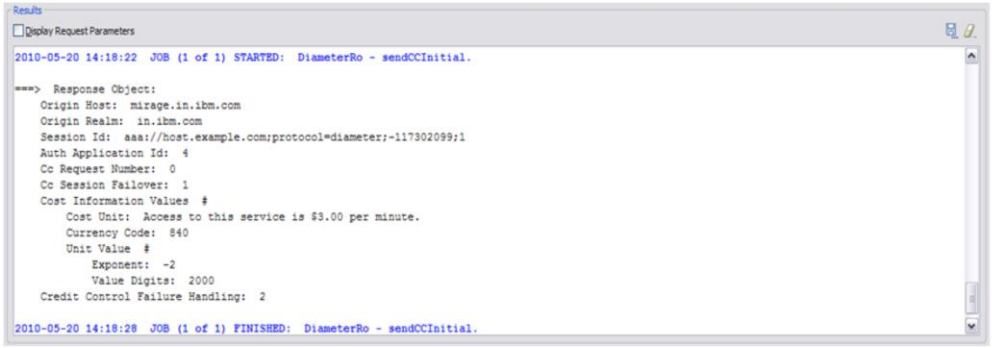
- Show Code Snippet button which shows the Java™ code that is built and used to run the request using the parameter values entered by the user. The code in this dialog can be copied using the Copy button and pasted in a Java class and reused for building a Java client for the request.

- Delete Requests button to delete any requests already in the batch.

Upon running the tests, moving the mouse up and down and selecting different requests shows the corresponding output in the results window. This is a useful feature, especially if the batch consists of a number of requests.

Diameter Ro client results

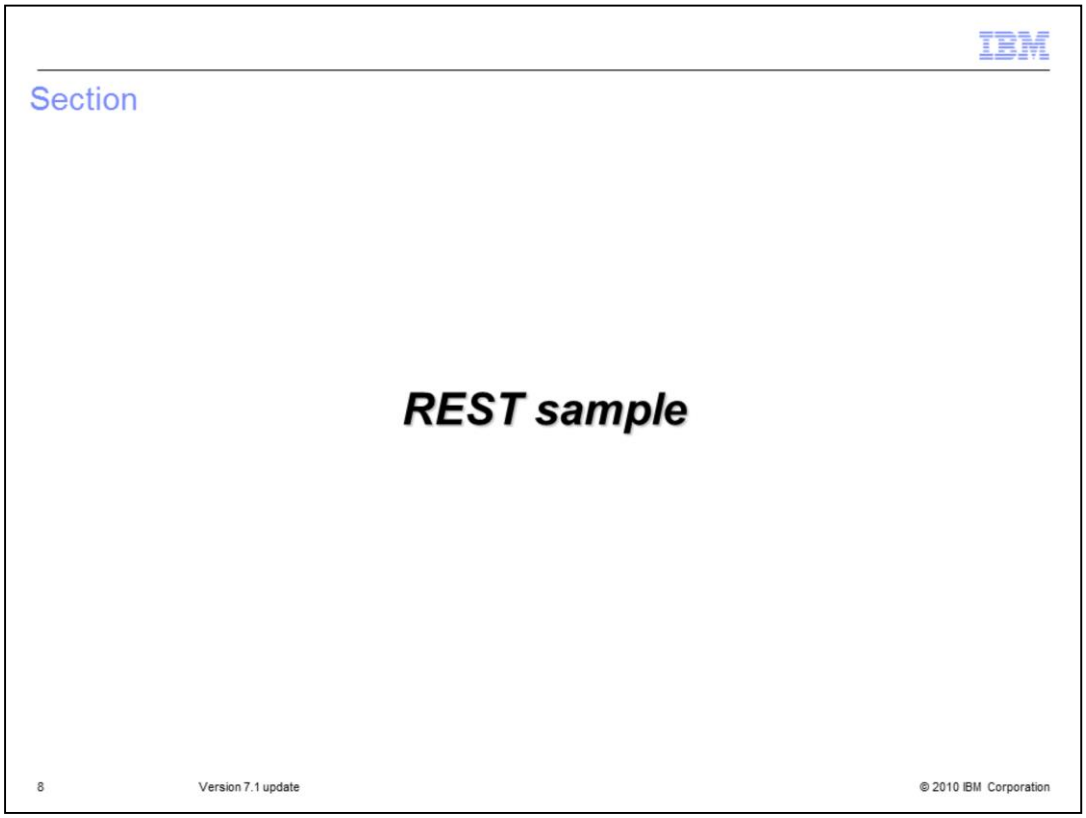
- Results displayed in the common Results section
  - Figure shows results of sendCCInitial() method
- Can be displayed with or without request parameters



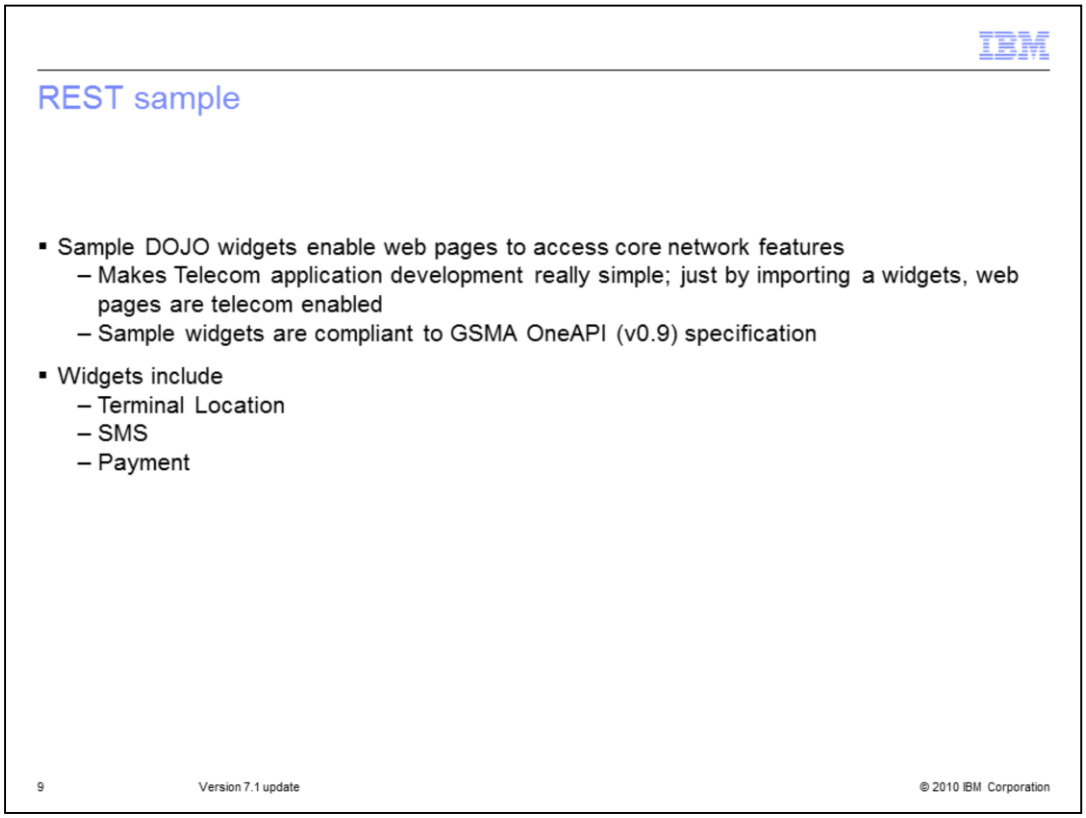
7 Version 7.1 update

The results of running single or batched Diameter Ro request are displayed in the common Results section.

The figure in the slide shows the results of a Diameter Ro request. The Response Object shows the service data returned for the sendCCInitial() method. The Results section is common for all the test clients and displays the results of all the client requests. The Results can be displayed with or without the request parameters. By default, the Display Request Parameters check box is checked. The results displayed can be saved to an external file using the save to file button and the contents can be cleared using the Clear button. You can change the font or color of the text by right clicking and selecting the Font menu item.

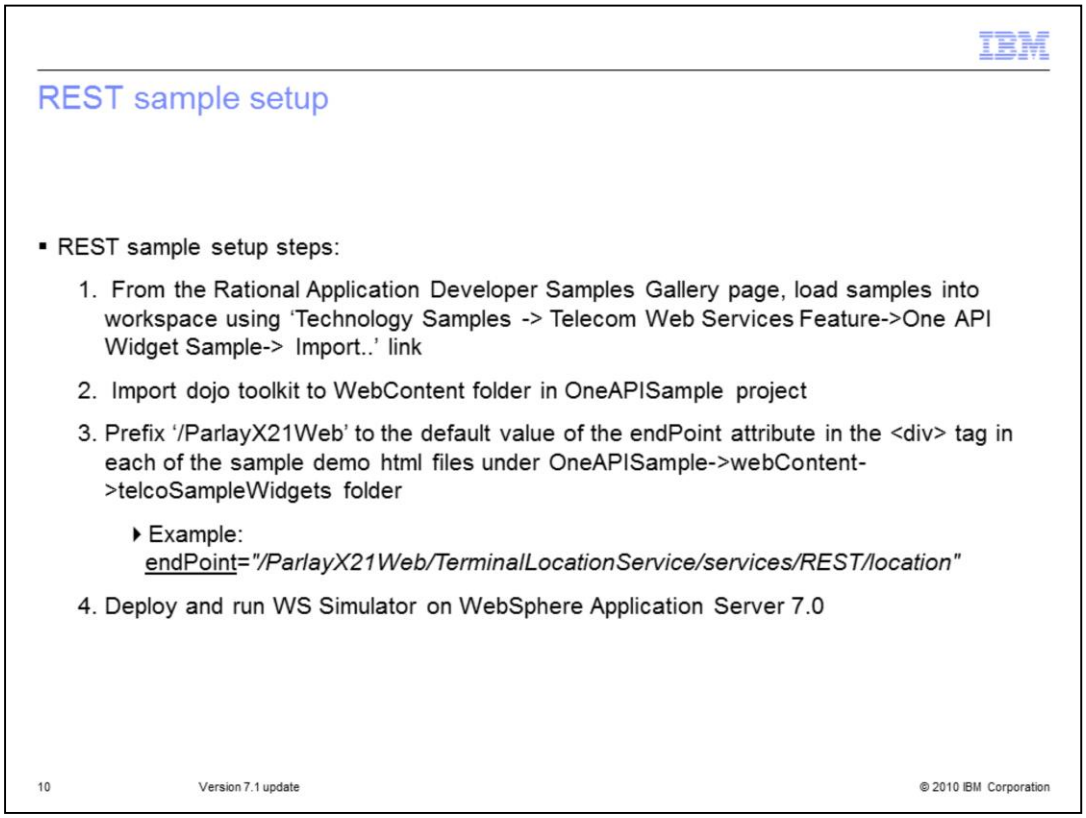


Moving on to the next topic of the agenda that is REST sample. The following slides gives an overview on REST sample which is added to the Telecom Web services feature in 7.1 release.



TWSS 7.1 provides a GSMA OneAPI specification v0.9 compliant, RESTful interfaces in addition to existing Web Service interfaces. To complement these, REST style sample widgets have been included in the Telecom Web Services feature of the toolkit. These REST sample widgets demonstrate the development of web pages to access network features. The REST sample widgets are built using the Dojo toolkit available in the Web Services Feature Pack for Rational® Application Developer. The sample includes widgets for Terminal location, SMS and Payment services.

If the Web Services Feature pack is installed, the Dojo libraries available in the feature pack can be used. If the feature pack is not installed, the Dojo toolkit can be downloaded from http://www.dojotoolkit.org.

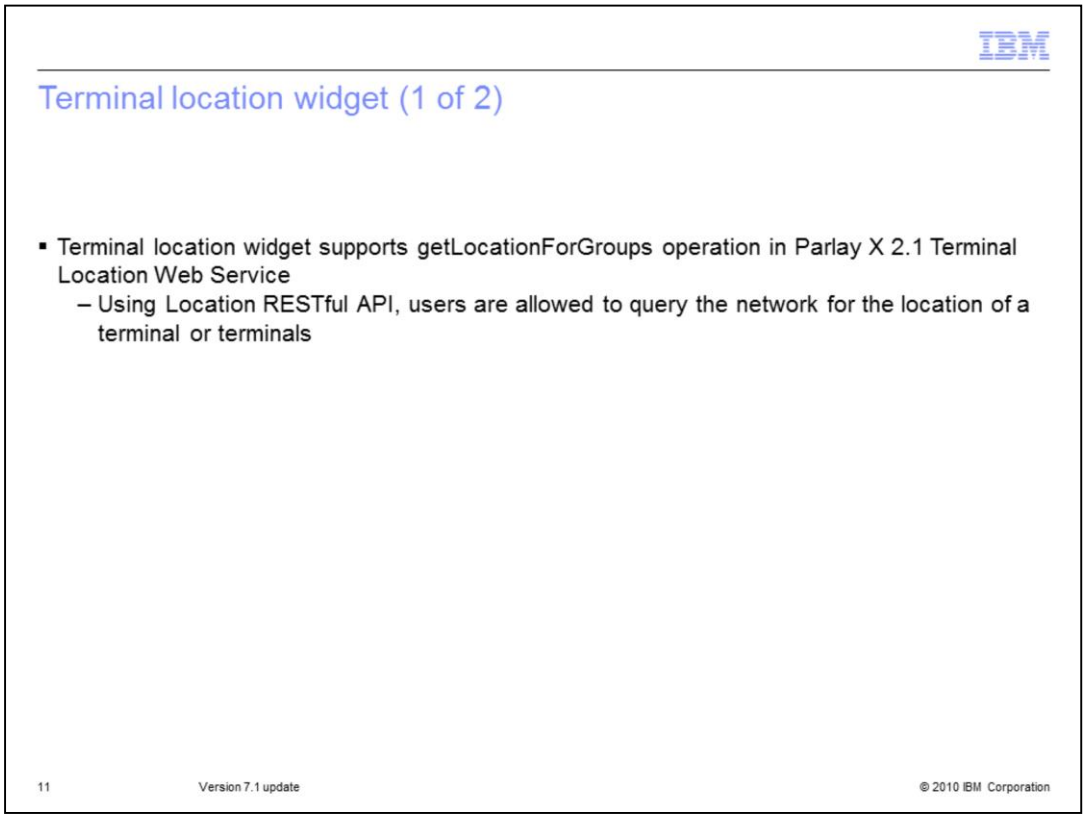


The setup process of the REST Samples involves these steps:

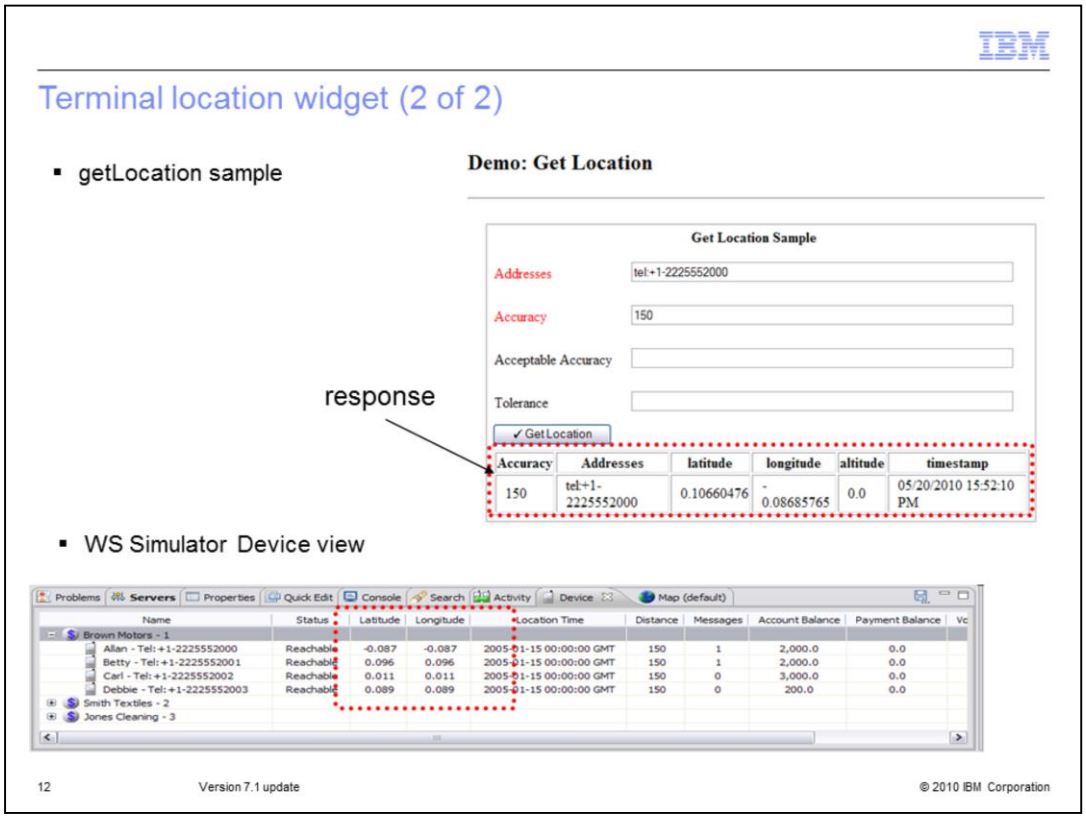
1. Load the samples into the Rational Application Developer workspace using the Import links from the Telecom Web Services Feature Samples page. The Telecom Web Services Feature Samples page can be accessed from the Help->Samples->Technology Samples->Telecom Web Services Feature-> Telecom Web Services Samples-> One API Widget Sample.
2. Review the prerequisite instructions by clicking on Setup Instructions link.
3. Import the sample.
4. Import dojo toolkit to WebContent folder in OneAPISample project.
5. Prefix '/ParlayX21Web' to the value of the endPoint attribute in the tag in each of the sample demo html files under OneAPISample->webContent->telcoSampleWidgets folder

   For example the endPoint of Terminal location should be *"/ParlayX21Web/TerminalLocationService/services/REST/location*
6. Deploy and run WS Simulator on WAS 7.0

Note: These OneAPI sample widgets can be tested on the Web services Simulator provided in the toolkit.



Terminal Location Web service provides operations for sending Terminal Location requests. The REST sample widget for Terminal Location provides getLocation interface to retrieve the location of a terminal or terminals



The figure in the slide shows the sample html page with Terminal Location widget.

The parameters required to send a getLocation request from the widget includes.

Addresses – indicates group of Telephone numbers. For example, 00441234123. The user may supply multiple address elements which needs to be separated by comma.

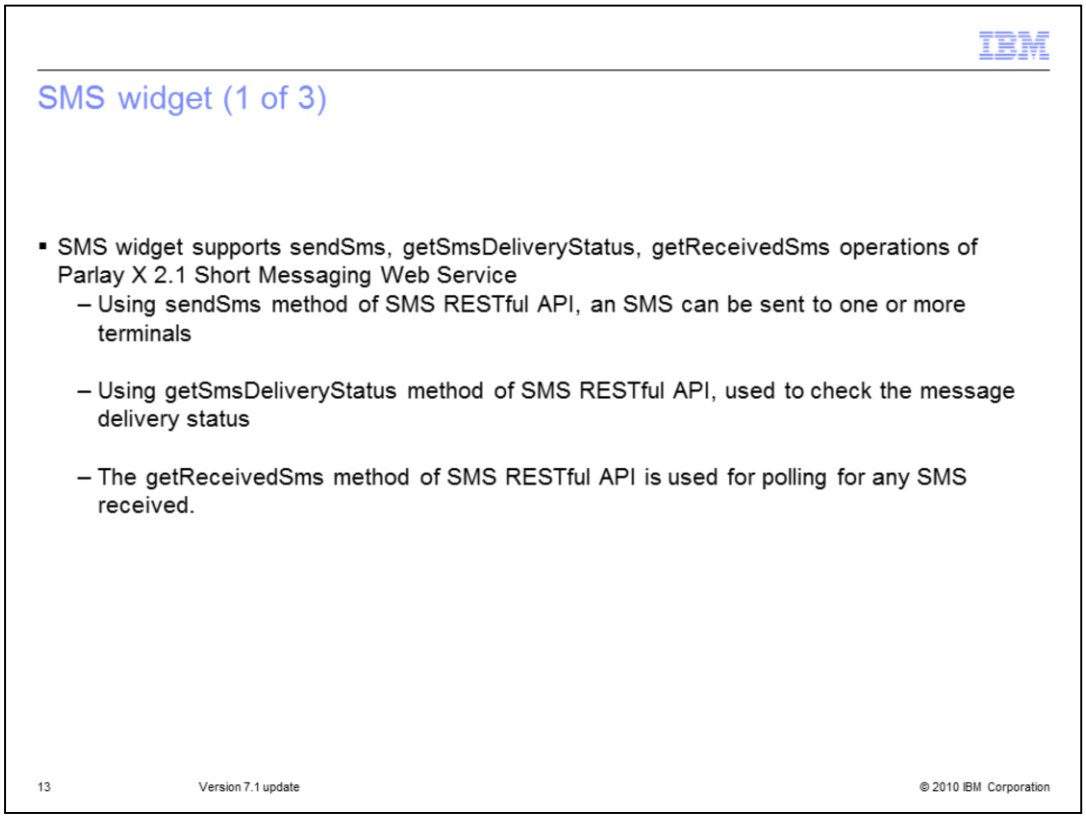
Accuracy – is the desired accuracy for the response in meters.

Acceptable Accuracy – is the limit acceptable to the requester in meters.

Tolerance – Indicates the priority of response time versus accuracy. Possible values are 'NoDelay', 'LowDelay' and 'DelayTolerant'.

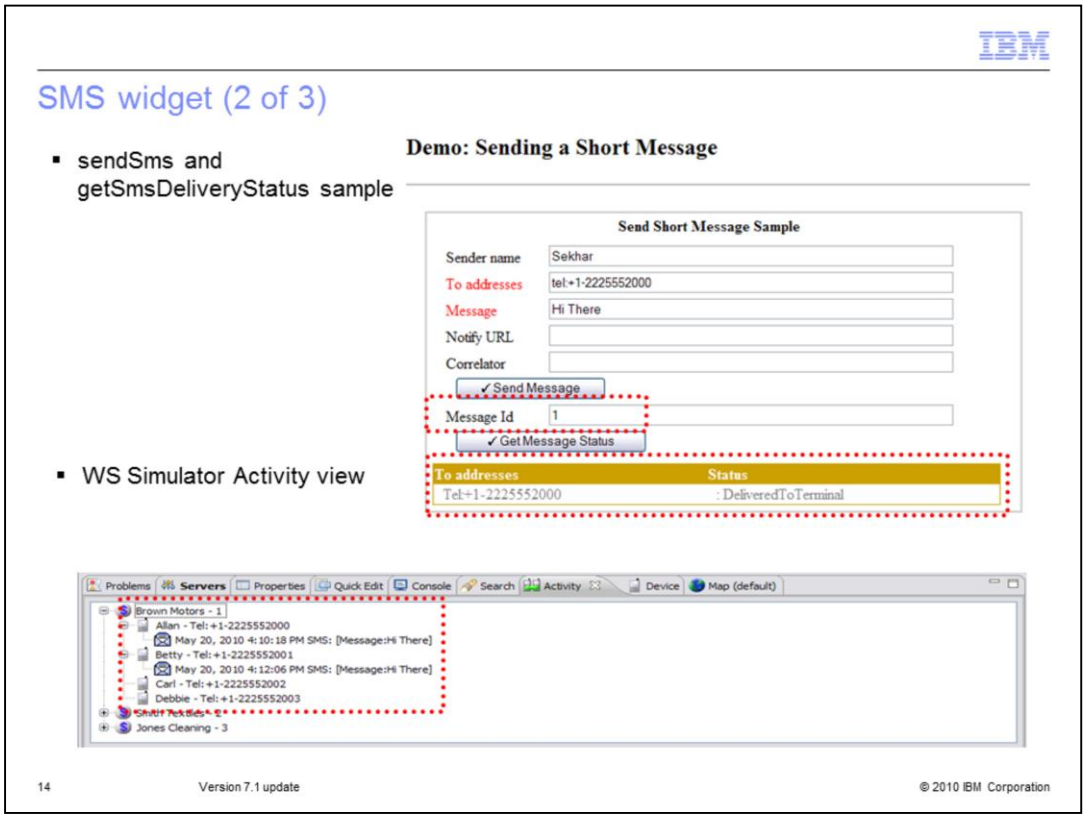
Out of the four parameters, Addresses and Accuracy are mandatory and the remaining two are optional parameters. After defining the parameters, the request should be submitted by clicking on Get Location button in the widget. The response consists of the terminal location details and are displayed in a table format in the same page.

The result can be cross verified with the corresponding terminal details from the Device view in the Web Service Simulator client as shown in the figure 2.



Parlay X SMS Web service provides operations for sending an SMS message to the network, monitoring the delivery status of a sent SMS message. Two sample widgets are available in REST sample for SMS application. One of the widget supports sendSms, getSmsDeliveryStatus operations and the other one supports getReceivedSms operation.

sendSms operation allows the user to send an SMS to one or more terminals and getSmsDeliveryStatus operation is useful to monitor the message delivery status. getReceivedSms can be used for polling for any SMS received



The figure 1 in the slide shows the sample html page with SMS widget which supports sendSms and getSmsDeliveryStatus operations. Parameters required to run this widget includes

Sender Name – is the name of the sender

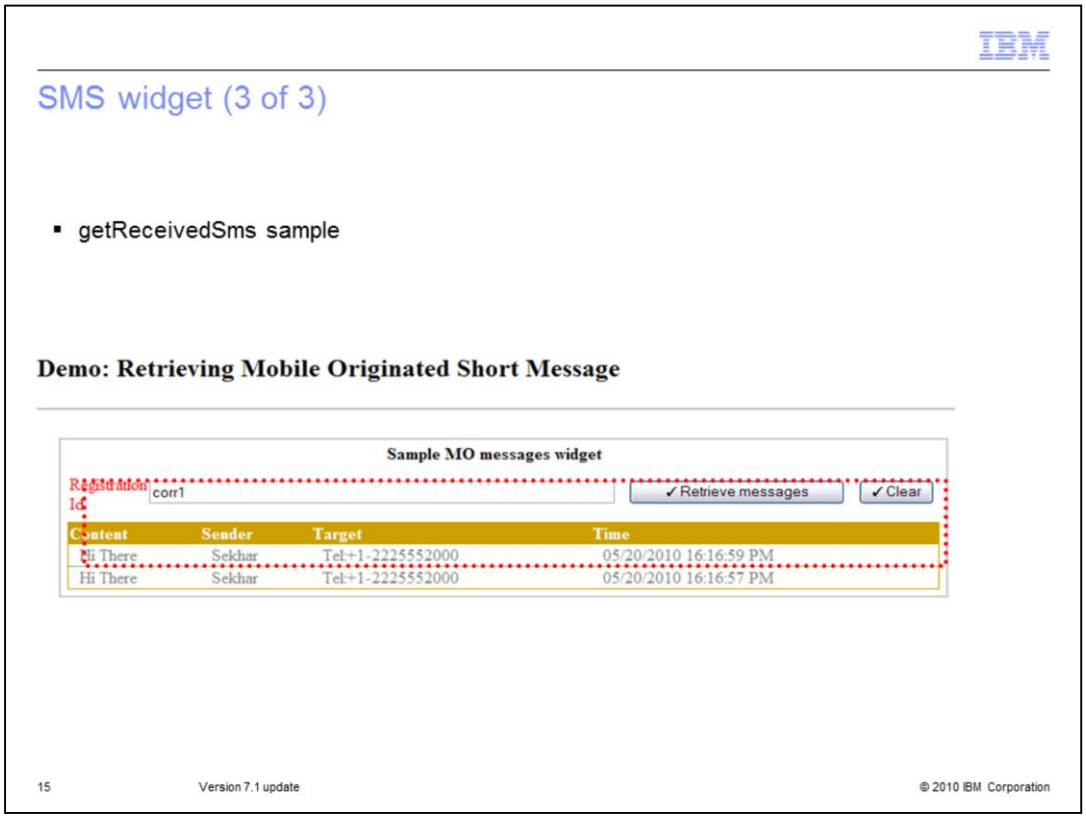
Addresses – indicates group of Telephone numbers. Ex. tel:+ 44799012122

Message – Is the message to be sent to the terminal

Correlator – indicates a unique identifier for the message

Notify URL – is the end point URL to receive the notification after message delivery

After defining all the mandatory and optional parameters, the sendSms request can be submitted by clicking on ‘Send Message’ button in the widget. A unique message id which identifies a specific SMS delivery request will be received as a response and it will be displayed in the ‘Message Id’ text field in the widget. The same message id is used as a parameter when getSmsDeliveryStatus request is submitted by clicking on ‘Get Message Status’ button. The responses for sendMessage and getSmsDeliveryStatus requests are highlighted with dotted rectangles in the figure 1. Once the message is being sent, the message status could be verified from the Web services simulator client Activity view. In the figure 2, the highlighted portion shows the message received by the terminal



The figure in the slide shows the sample html page with second SMS widget which supports getReceivedSms interface. Before running this sample,

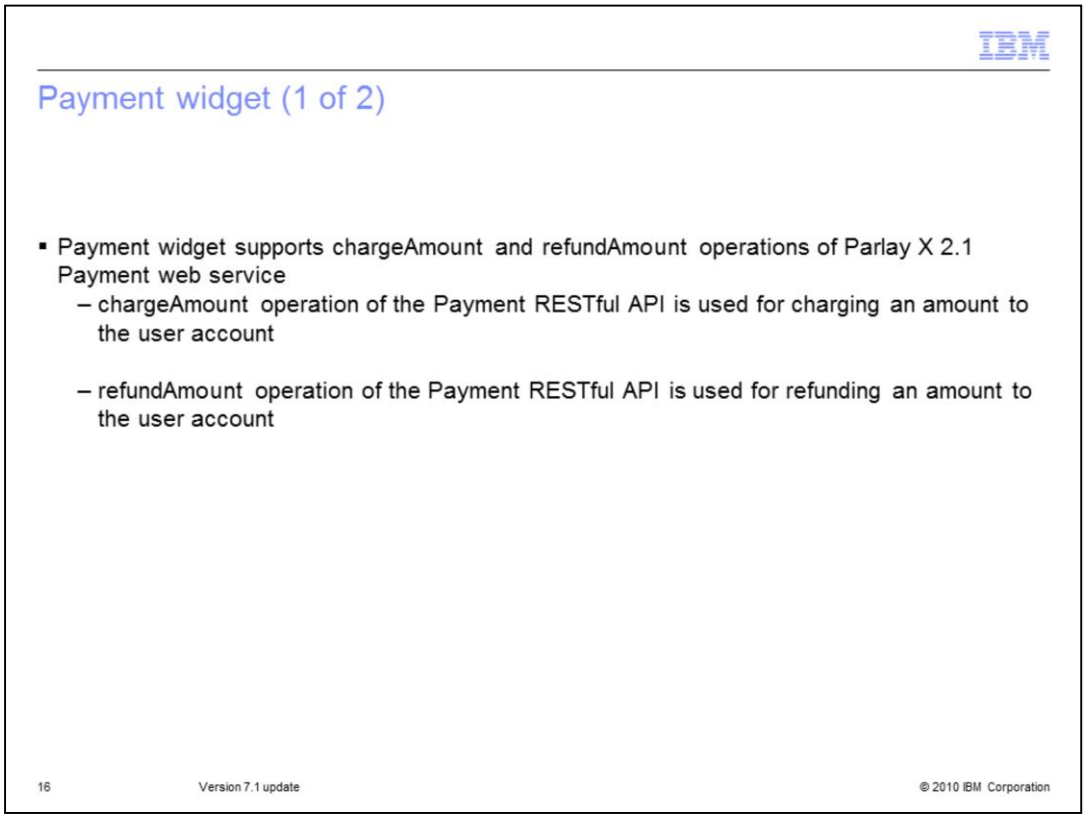
1. A notification should be created on the target short ID with a correlator and criteria and

2. Send few Mobile Originated messages to this ID, matching the criteria used for starting the notification (However, to run this sample, a normal telephone number listed in the WS configuration file can also be used as the target to create a notification and to send messages. For creation of notifications, Telecom Samples can be used from Telecom Web Service feature.)

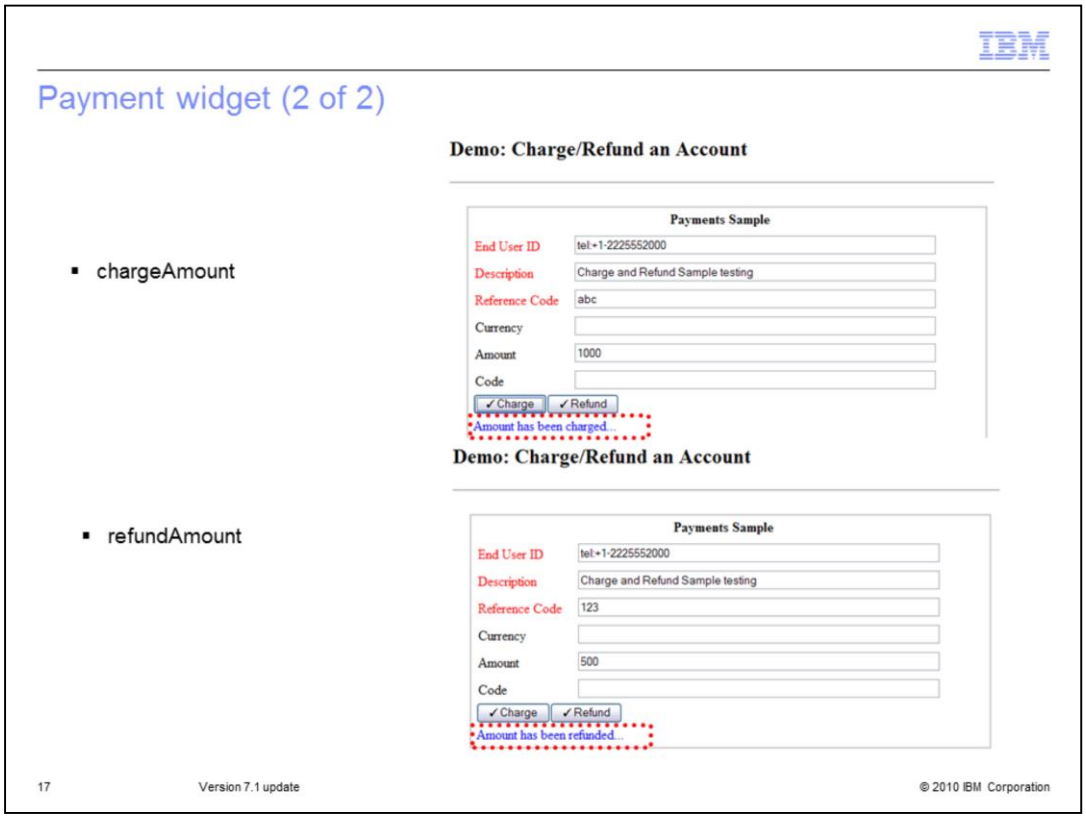
3. Use the supplied Widget to retrieve the messages sent for the correlator specified. The required parameter to run this sample widget includes

Registration ID – Identifies the off-line provisioning step which enables the application to receive notifications of the SMS reception. This ID should be same as the correlator specified during notification creation.

The getReceivedSms request retrieves all SMS messages received that fulfill the criteria identified by correlator. The Web Service operation returns only the list of SMS messages received since the previous invocation of the same method, (that is, each time the method is executed the messages returned are removed from the server)



Parlay X Payment Web service allows Telecom Service Provider to easily enable an application to charge an amount against a user account or place money into a user account The sample widget for Payment application supports chargeAmount and refundAmount operations. chargeAmount operation is used for charging an amount to the user account and refundAmount operation is used for refunding an amount to the user account



The figure in the slide shows the sample html page with Payment widget. The required parameters to run the Payment widget includes

User ID – Is an unique id of the customer.

Description – Information that appears on the bill.

Currency – The amount of currency.

Amount – is the amount to be charged.

Code – The charging code of the contract.

Reference code – Textual information which uniquely identifies the request.

The parameters highlighted in Red color are mandatory and rest of the parameters are optional. Since, both chargeAmount and refundAmount operations requires same set of parameters, both of these service operations can be invoked from the same widget by clicking either ‘Charge’ button or ‘Refund’ button. The result string gets displayed in the bottom of the widget as shown in the two figures in highlighted rectangles.

Trademarks, disclaimer, and copyright information