

Leading the **Integration** Revolution

Your business problems have changed.
Why hasn't your integration solution?



ADEPTIA

Use Case: Publishing an orchestration as a REST API

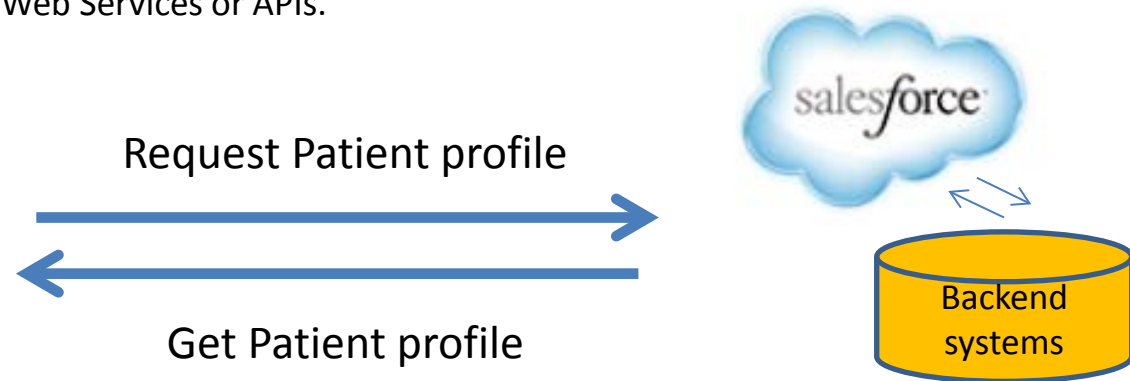
High-level scenario

Client sends a request via RESTful API to get a Patient profile by sending a Patient ID and receives a derived result back from a mashup of different services that are executed as part of an orchestration.

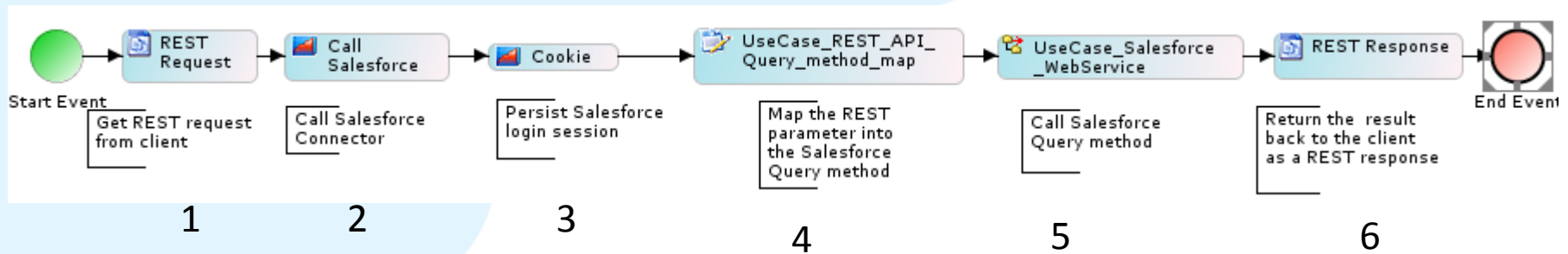
In this example the important service that is being executed in the orchestration is Salesforce SOAP Web Service which takes the incoming REST Request parameter and queries the matching account in Salesforce CRM and sends the result. You can add additional services such as pulling data from a database and other backend systems and merging them into the response. The result is sent back as a RESTful API response to the client.

Idea of providing a simple public-interfacing REST API allows your external clients to request and get data from your cloud and enterprise applications without having to go through the complexity of learning how to interact with these system's native APIs. Orchestrations published as a REST API can be used as a wrapper with simple methods that can be consumed by your external clients to interact with your applications. The orchestration would contain multiple system connections and data mappings needed to derive the result requested by the client.

Goal of this example is to use it as a working template to expand on your orchestration steps and include other system interactions with databases, Web Services or APIs.



Technical design



For this orchestration we are going to call Salesforce when the process receives a REST request from the client. We have published the orchestration as a public facing REST API that can be called by any client application.

The orchestration has simple steps to login to Salesforce, find the Account that matches the Patient ID coming from the request and then returns the response back to the client. The response format can be in XML, JSON, text etc.

Description of the process design:

File event triggers the process flow on arrival of new or modified file.

1. “REST Request” is the HTTP REST Request that is sent by the client
2. “Call Salesforce connector” is a Call action that executes the Salesforce Connector subprocess flow
3. “Cookie” is a context variable that assigns the serverURL of the session to the Web Service activity (#5)
4. “REST_API_Query_method_map” is a mapping where we assign the incoming REST parameter value to the query
5. “Salesforce_WebService” is a Salesforce Web Service SOAP call that executes the Query method
6. “REST Response” returns the response from the Salesforce query back to the client as REST response (in XML)

Let's run the REST API

Run-time: Client calls REST API and gets the result

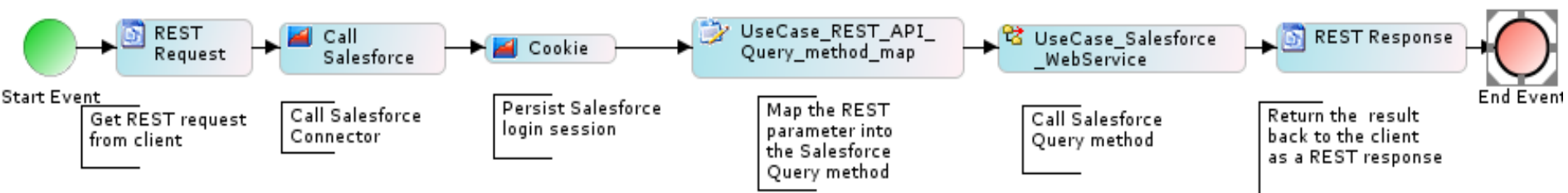
The screenshot shows a REST client interface with the following details:

- URL: `http://localhost:8080`
- Full Path: `[/adeptia/publishProviderByRest/getpatient/profile]`
- Request Panel (left):
 - Table with columns 'Name' and 'Value':

Name	Value
id	JoeC
 - Label: "Request" with a red arrow pointing to the table.
- Response Panel (right):
 - XML content:

```
<queryResponse xmlns="urn:enterprise.soap.sforce.com" xmlns:sf="urn:subject.enterp">
  <result>
    <done>true</done>
    <queryLocator xsi:nil="true"/>
    <records xsi:type="sf:Account">
      <sf:Id xsi:nil="true"/>
      <sf:BillingCity>Chicago</sf:BillingCity>
      <sf:BillingStreet>21 Main St</sf:BillingStreet>
      <sf:Description>New Patient registration</sf:Description>
      <sf:Name>JoeC</sf:Name>
      <sf:Phone>(312) 009-1111</sf:Phone>
    </records>
    <size>1</size>
  </result>
</queryResponse>
```
 - Label: "Response" with a red arrow pointing to the XML content.

Request executes this orchestration and the result is sent back to the client



Run-time: Monitor the progress and the final status of the REST API Request

Run time can be monitored by going to Monitor > Dashboard > Process Flow

This dashboard shows all the run-time executions of the orchestrations that are triggered from the REST API requests. You can look at the actual request, response and the status of the orchestration.

The screenshot displays the Adeptia Monitor interface. The top navigation bar includes 'Home', 'Develop', 'Monitor', and 'Administer'. Below this, there are dropdown menus for 'My Tasks', 'My Documents', 'Dashboard', 'Reports', 'History', and 'My Solutions'. The main content area is titled 'Dashboard > Process Flows'. On the left, a 'Process Flows List' shows 'UseCase_REST_API_Orchestration' selected, with a red arrow pointing to it. The main panel, 'Process Details', shows the process name and a description: 'UseCase_REST_API_Orchestration (Process that gets the client request and derives the result from d...)'. It features two bar charts: 'Status' and 'Last 10 Days'. The 'Status' chart shows 0 instances in 'Queued' and 'Running' states, 10 in 'Failed', and 5 in 'Executed'. The 'Last 10 Days' chart shows 0 instances from 03 Nov to 09 Nov, 30 on 11 Nov, and 15 on 12 Nov. Below the charts is a table titled 'Process > Process Flow' with a 'Quick Search' field. The table has columns for 'Description', 'Status', 'Started By', 'Start Time', 'Duration', and 'Action'. It lists three entries, all with 'Executed' status and a duration of 6 seconds.

Description	Status	Started By	Start Time	Duration	Action
Process that gets the client request and derives the result from different services...	Executed	demouser	11/12/2013 13:05:28	6 sec	⋮
Process that gets the client request and derives the result from different services...	Executed	demouser	11/12/2013 11:45:35	6 sec	⋮
Process that gets the client request and derives the result from different services...	Executed	demouser	11/12/2013 11:45:10	6 sec	⋮

Let's build the process flow

Design methodology



Design a high level process flow



Configure all the activities in the process flow



Publish the process flow as a REST API

Design: First let's create a Top level design of the solution using the Process Designer

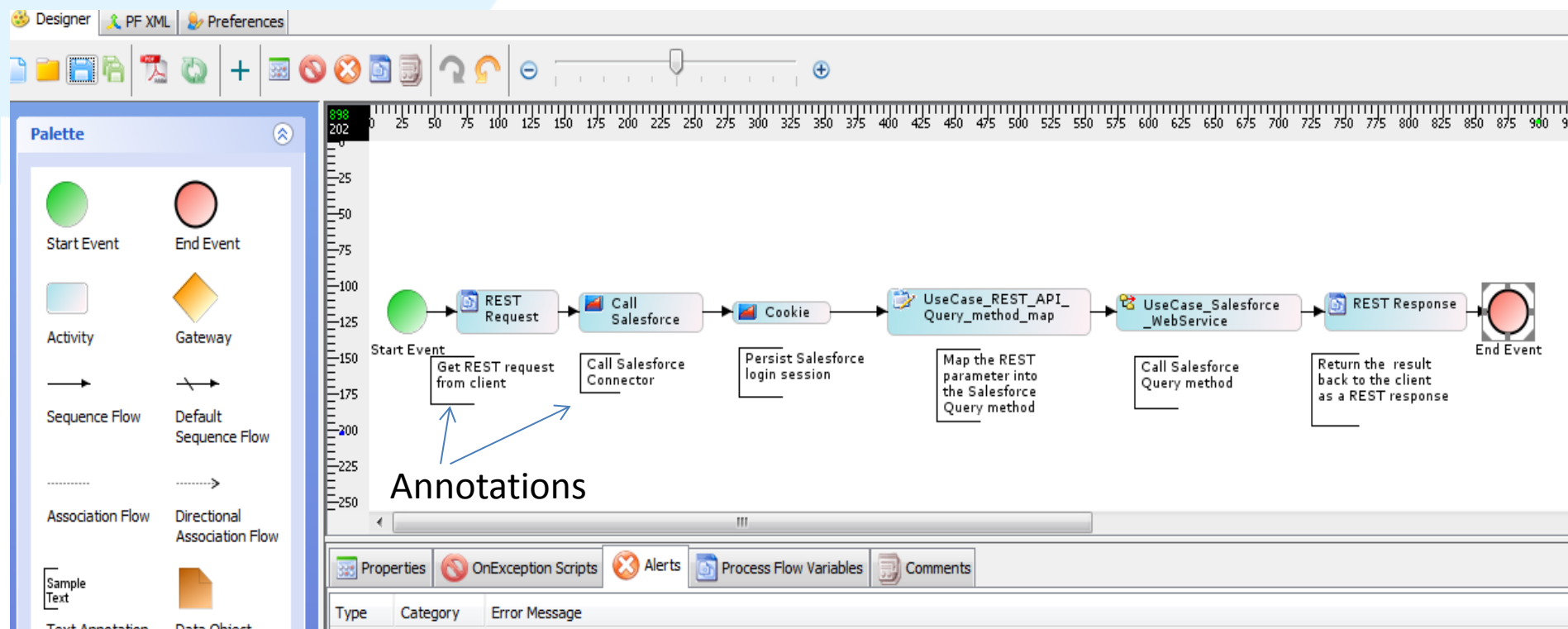
Go to Develop > Process > Process Flow

Create new and in the designer design a top-level flow by pulling the icons from the palette area.

Refer to the reference video and the annotations for each activity to see how to design a process flow.

Each activity performs a discreet function such as getting request, mapping, sending response.

The Salesforce connector is already pre-built and we are reusing it in this process flow.



Configure: Let's map the REST API request to Salesforce query method

Go to Develop > Data Transformation> Mapping

In this mapping we are mapping the "id" into the Query field of the Salesforce Query method.

We created local variables to store the value and build the query string that is passed to the target element.

varString is a local variable that gets the value form "id" parameter.

varSelect is a local variable that has a SOQL string as shown below

varQuery concatenates the "id" and the query string together. This variable is mapped to the target field.

The screenshot shows a mapping tool interface with the following components:

- Structure_of_source_XSD:** A tree view showing a hierarchy: Root (inputContextSchema) > Record > id.
- Structure_of_destination_XSD:** A tree view showing a hierarchy: tns:query (query) (wsSchema_UseCase_Salesforce_WebService) > tns:queryString (string) (M, LV) ✓.
- Local Variables:** A list of variables defined in the mapping:
 - varString (\$Input_inputContextSchema/Root/Record/id) ✓
 - varSelect ('SELECT Name, AccountNumber, BillingStreet, BillingCity, AnnualRevenue, Phone, Description FROM Account WHERE Name =') ✓
 - varQuery (concat(\$varSelect , \$apos,\$varString,\$apos)) ✓
- Mapping Rules:** A list of rules, with \$varQuery ✓ selected.
- Current Element:** tns:queryString
- Tools:** A toolbar at the top with icons for navigation and editing.
- Bottom Panel:** A panel with tabs for Textual Rules, Local Variables, Properties, and Mapping Rules.

Deploy: Publish Process Flow as a REST API

Create Provider activity by going to Develop > Services > Web Services > Provider

Here we are publishing our process flow as RESTful Web Service using Get method. We also defined the parameter name as “id”.

The screenshot shows the Adeptia web interface. The navigation menu includes Home, Develop (checked), Monitor, and Administer. The breadcrumb trail is Services > Web Services > Provider (checked). Below the breadcrumb, there are actions: Synchronize, Delete, and Create New. A table lists the providers, with 'getpatient' selected. The 'Edit Web Service Provider: getpatient' form is shown below, with fields for Name, Description, Character Set Encoding, Publish Type (REST selected), Resource End Path, and Process Flow Name. A table for Resource Parameters is also visible, with the first row having Name 'id' and Style 'Query'. At the bottom, there is a table for Method Parameters with Method Type 'GET'.

Services > Web Services > Provider

Synchronize Delete + Create New Quick Search

Name	Description	Owner	Style	WSDL
getpatient	publish orchestration as a RESTful API	de...	REST	View

Edit Web Service Provider: getpatient

Name: getpatient ✓

Description: publish orchestration as a RESTful API

Character Set Encoding: ISO-8859-1 Refresh

Publish Type: SOAP REST ✓

Resource End Path: /profile ✓

Process Flow Name: UseCase_REST_API_Orchestration

#	Name	Value	Style
1	id ✓		Query
2			Query
3			Query
4			Query

No. of Rows: 1 at Position 5 Add Row Remove Row

Add Method

#	Method Type	Action
1	GET ✓	Edit Delete

Save

Managing the activities created for this use case

You can manage and view all the activities of this use case by going to Develop > Projects
Click on the project named "UseCase_REST_API_Orchestration" and it will show all the activities configured for this process under its related categories. You can open any category to view its activities.

The screenshot shows a software interface with a navigation bar at the top containing 'Home', 'Develop', 'Monitor', and 'Administer'. Below this is a secondary navigation bar with 'Getting Started', 'Projects', 'Solutions', 'Process', 'Services', 'Events', 'Web Forms', and 'Reports and Dashboards'. The main content area is titled 'Projects' and displays a list of activities for the project 'UseCase_REST_API_Orchestration'. The list is organized into categories: Data Mapping (1), Process Flow (1), and Web Service Provider (1). Each activity row includes a checkbox, a name, a description, an owner, a modified date, and an action menu.

<input type="checkbox"/>	Name	Description	Owner	Modified	Action
Data Mapping (1)					
<input type="checkbox"/>	UseCase_REST_API_Query_method_map	Map Query method of Salesforce Web Service	demouser	11/12/13 1...	☰
Process Flow (1)					
<input type="checkbox"/>	UseCase_REST_API_Orchestration	Process that gets the client request and derives the result from differe...	demouser	11/12/13 1...	☰
Web Service Provider (1)					
<input type="checkbox"/>	getpatient	publish orchestration as a RESTful API	demouser	11/12/13 1...	☰

Leading the **Integration** Revolution

Your business problems have changed.
Why hasn't your integration solution?

Thank You!



ADEPTIA