IBM Technical Summit

Stay ahead.

IBM

Enterprise Service development for Mobile Devices using Rational Software Architect and IBM Worklight

Manoj Paul
Senior Staff Software Engineer, IBM
manojpaul@in.ibm.com

Agenda

- Introduction
- Developing Enterprise Services
- Enabling Mobile Access
- Demo
- Summary

Introduction - Mobile Apps – Top Challenges

Creating rich, yet cost-effective mobile apps in a fragmented technological landscape.

Connecting the enterprise back-end services in a secure and scalable manner

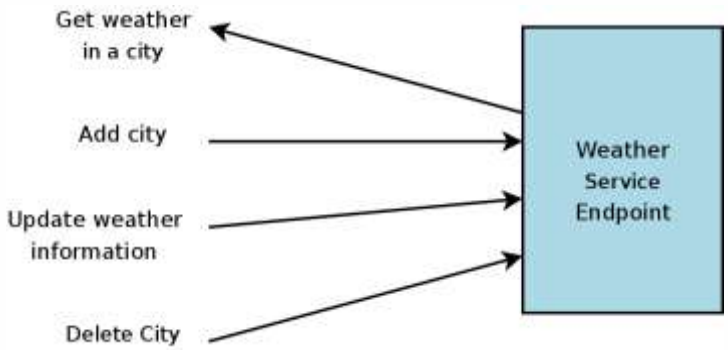
Controlling the growing portfolio of applications deployed “in the wild”

Enterprise Service Development

- How do you develop back-end services?
  - Choice of Architecture
  - Choice of Implementation Framework
- How do mobile-enable these services?

Technology - REST : Representational State Transfer (REST)

- **REST defines a set of architectural principles for designing Web services**
  - Focus on resources, including how resource states are addressed and transferred over HTTP.
- **A simpler alternative to SOAP- and Web Services Description Language (WSDL)-based Web services**
- **Has gained widespread acceptance across the Web**
  - Adoption of REST by mainstream Web 2.0 service providers—including Yahoo, Google, and Facebook
- **REST Web service follows four basic design principles**:
  - Use HTTP methods explicitly.
  - Be stateless.
  - Expose directory structure-like URIs.
  - Representation of resource state



REST Concepts

- REST design principle establishes a one-to-one mapping between create, read, update, and delete (CRUD) operations
- REST suggests the design of web services be stateless
- Expose directory structure-like URIs
- Resource Representation

Why RESTFul Services for Mobile?

- Lightweight programming model
  - Based upon HTTP GET, POST, DELETE …
- Easily consumable
- Supports multiple data representations
  - JSON, XML
  - Easier to consume in Mobile frontends

Framework - JAX-RS

- JAX-RS: Java API for RESTful Web Services provides Java API for creating REST Services
- JAX-RS uses annotations to simplify the development and deployment of web services
  - @Path, specifies the relative path for a resource class.
  - @GET, @PUT, @POST, @DELETE, specifies the HTTP request type of a resource method.
  - @Produces, specifies the returned MIME media types
  - ....

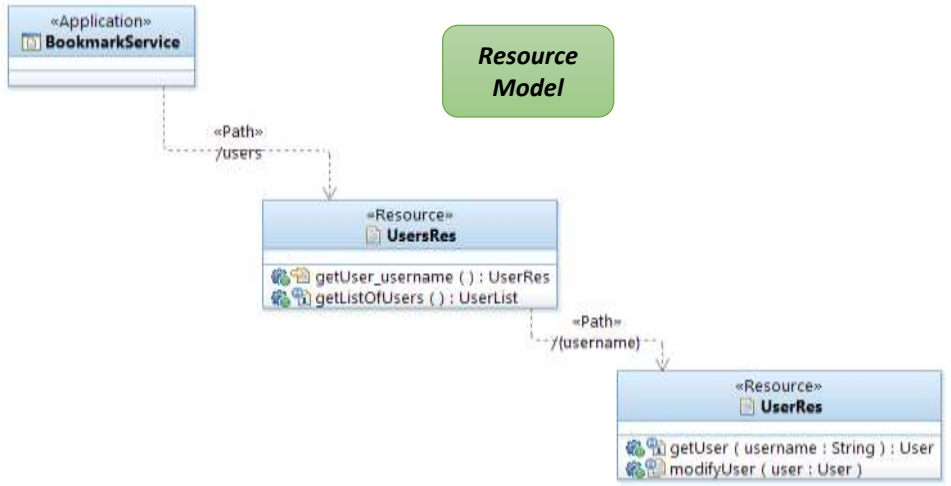
```
@Path("widgets")
@Produces("text/plain")
public class WidgetsResource {

        @GET
        @Path("offers")
        public WidgetList getDiscounted() {

        }

        @Path("{id}")
        public WidgetResource findWidget(@PathParam("id") String id) {
                return new WidgetResource(id);
        }
}

public class WidgetResource {
        public WidgetResource(String id) {        }
        @GET
        public Widget getDetails() {       }
}
```

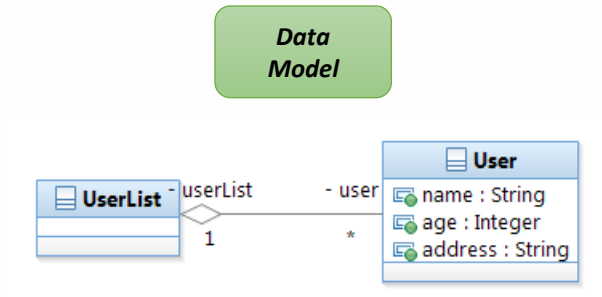
Rational Software Architect – MDD for RESTFul Services

- What Rational Software Architect provides
  - Modeling of RESTful Applications
    - Resource Modeling
    - Data Modeling
      - Using JAXB to support XML and JSON data
    - Scenario Modeling
      - With HTTP Header and Error Code support

- Code-generation / Reverse Engineering for JAX-RS based Server-side implementations

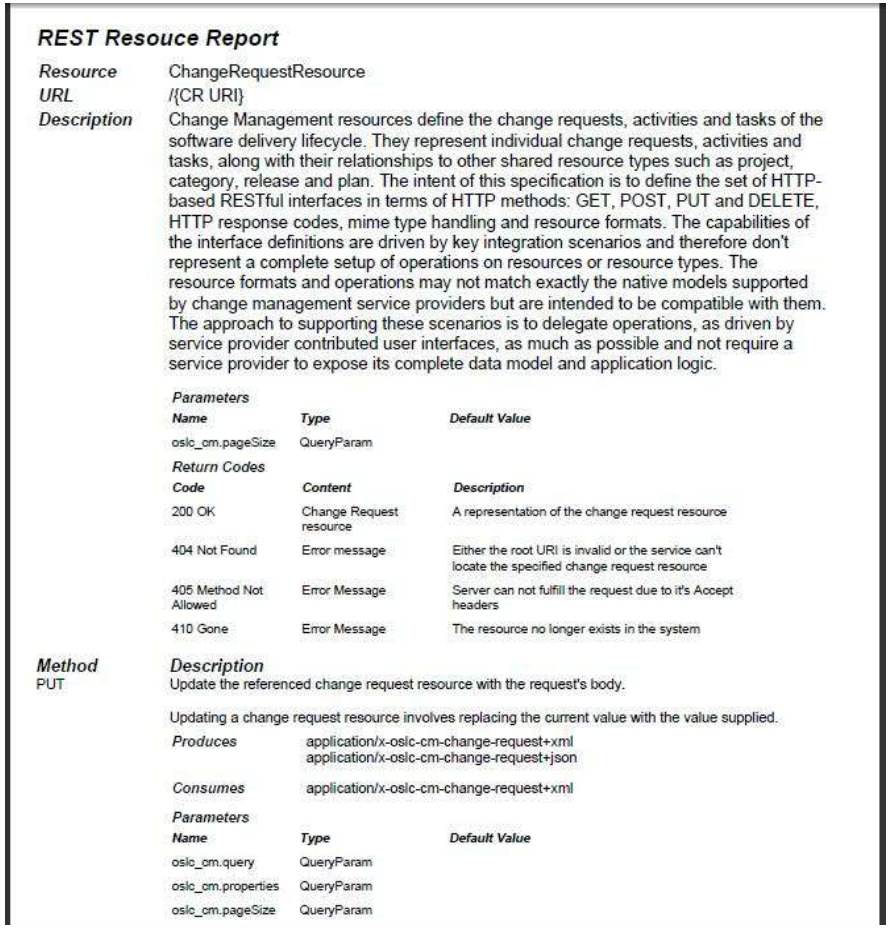
Rational Software Architect Support – RESTFul Modeling





BIRT reports for REST services

- A BIRT report will be provided which will list out details for all Resources in a model.



REST Resouce Report

Resource	ChangeRequestResource
URL	/{CR URI}
Description	Change Management resources define the change requests, activities and tasks of the software delivery lifecycle. They represent individual change requests, activities and tasks, along with their relationships to other shared resource types such as project, category, release and plan. The intent of this specification is to define the set of HTTP-based RESTful interfaces in terms of HTTP methods: GET, POST, PUT and DELETE, HTTP response codes, mime type handling and resource formats. The capabilities of the interface definitions are driven by key integration scenarios and therefore don't represent a complete setup of operations on resources or resource types. The resource formats and operations may not match exactly the native models supported by change management service providers but are intended to be compatible with them. The approach to supporting these scenarios is to delegate operations, as driven by service provider contributed user interfaces, as much as possible and not require a service provider to expose its complete data model and application logic.

Parameters

Name	***Type***	***Default Value***
oslc_cm.pageSize	QueryParam	

Return Codes

Code	***Content***	***Description***
200 OK	Change Request resource	A representation of the change request resource
404 Not Found	Error message	Either the root URI is invalid or the service can't locate the specified change request resource
405 Method Not Allowed	Error Message	Server can not fulfill the request due to it's Accept headers
410 Gone	Error Message	The resource no longer exists in the system

Method	***Description***
PUT	Update the referenced change request resource with the request's body.

Updating a change request resource involves replacing the current value with the value supplied.

Produces	application/x-oslc-cm-change-request+xml application/x-oslc-cm-change-request+json
Consumes	application/x-oslc-cm-change-request+xml

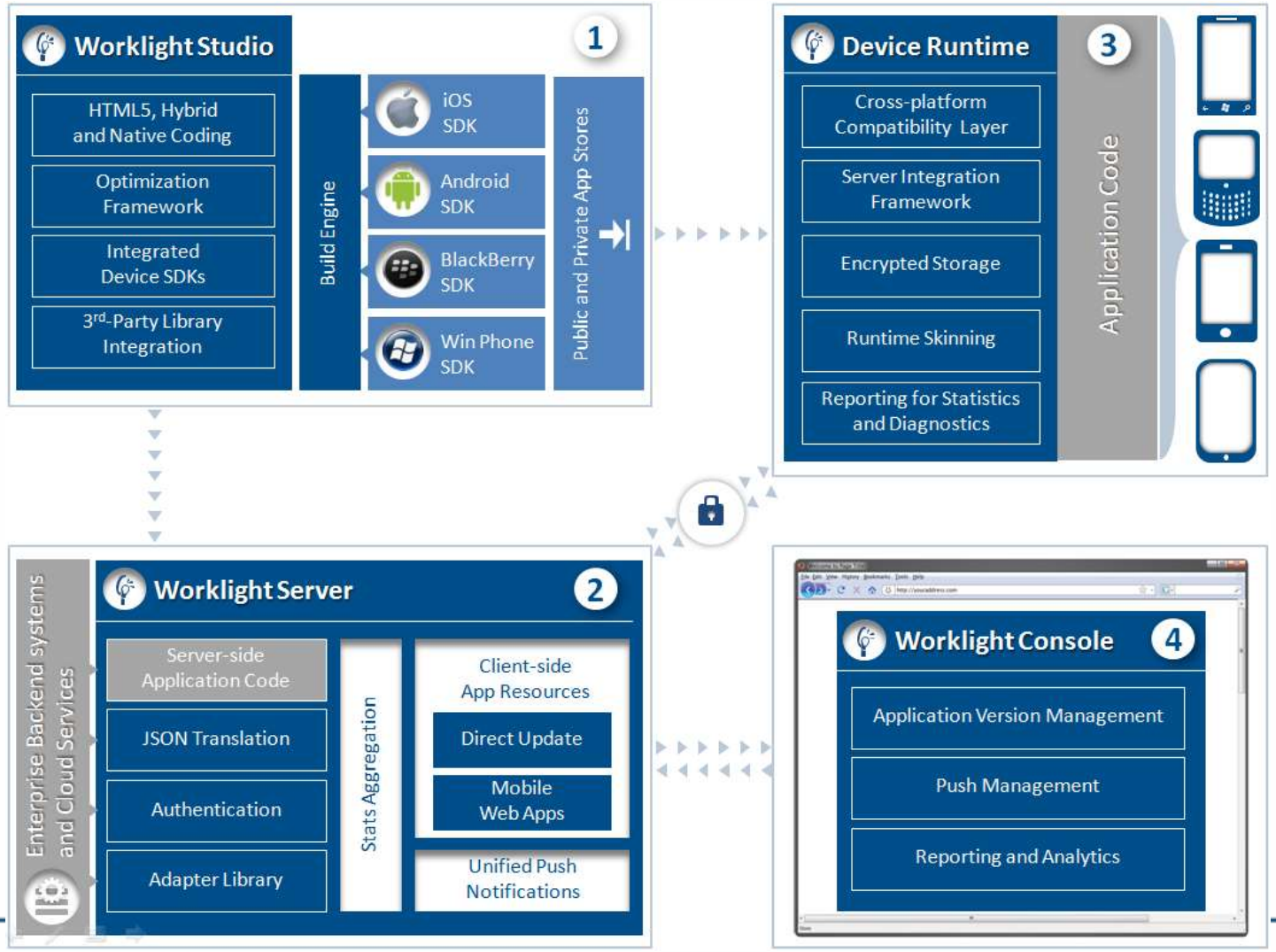
Parameters

Name	***Type***	***Default Value***
oslc_cm.query	QueryParam	
oslc_cm.properties	QueryParam	
oslc_cm.pageSize	QueryParam	

Worklight Introduction

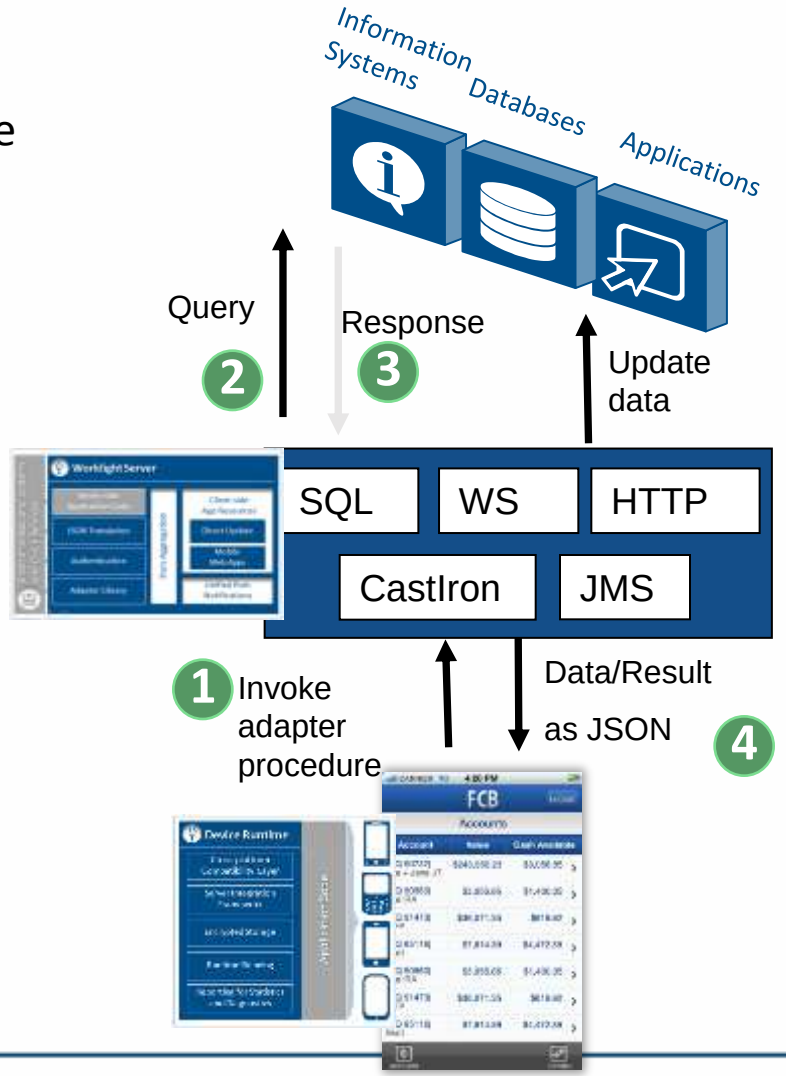
Worklight is an open, complete and advanced mobile application platform for HTML5, hybrid and native apps.

Worklight Architecture



Worklight Adapters

- An Adapter is a transport layer used by the Worklight Platform to connect to various back-end systems.
- Adapters are used for:
  - Retrieving information
  - Performing actions
- Out of the box:
  - HTTP Adapter
    - RESTFul and SOAP
  - SQL Adapter
  - Cast Iron Adapter



Rational Software Architect Support - Worklight

- What Rational Software Architect provides
  - Code-generation for Worklight based applications from RESTFul Models
    - Worklight Server-side Adapter
    - Client-side stub code
      - JavaScript functions to make calls to Server Side Adapter
      - Isolates Front-end development
- Shell-shared w/ Worklight Studio for Development

```
function getUsers_Users() {

        pathURL = '/users';
        var input = {
            method : 'get',
            returnedContentType :

'application/atom+xml',
            path : pathURL
        };
return
WL.Server.invokeHttp(input);
}
```

Rational Software Architect Support - Worklight

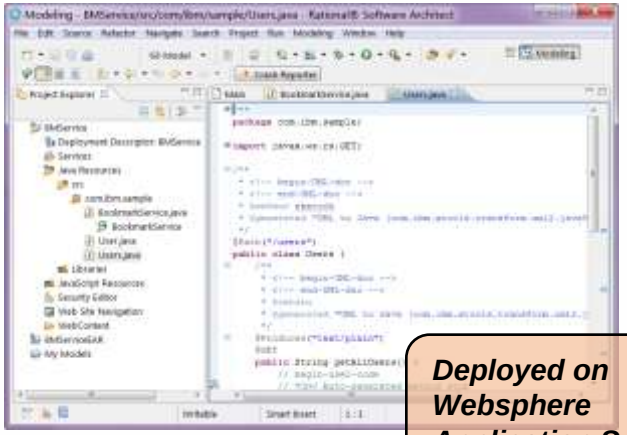
- Exposing existing RESTFul Services via Worklight
  - Reverse-engineer existing RESTFul Applications (JAX-RS based)
    - Generate Worklight Server-side Adapter
    - Generate Client-side stub code
  - Model RESTFul Service (non-JAX-RS based)
    - Generate Worklight Server-side Adapter
    - Generate Client-side stub code

Rational Software Architect Support Overview

1 *Model RESTFul Service in RSA*

2 *Generate JAX-RS based Web Service*

3 *Generate JAXB classes from Data Model*

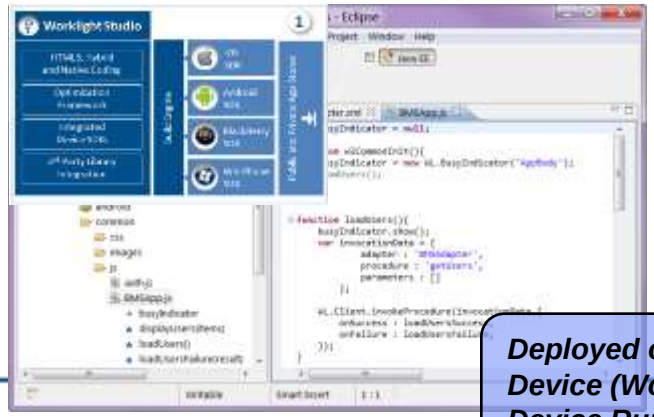


Deployed on Websphere Application Server

4 *Generate Worklight Adapter*

Deployed on Worklight Server

5 *Generate Worklight Client Stub*



Deployed on Mobile Device (Worklight Device Runtime)

Summary

- Enterprise Services Development for Mobiles
  - RESTFul Services / JAX-RS
  - Model-driven Development with Rational Software Architect
  - Mobile enablement using Worklight

- Links
  - http://www-01.ibm.com/software/rational/products/swarchitect/
  - http://www-01.ibm.com/software/mobile-solutions/worklight/

- Questions

www.ibm/software/rational

© **Copyright IBM Corporation 2012. All rights reserved.** The information contained in these materials is provided for informational purposes only, and is provided AS IS without warranty of any kind, express or implied. IBM shall not be responsible for any damages arising out of the use of, or otherwise related to, these materials. Nothing contained in these materials is intended to, nor shall have the effect of, creating any warranties or representations from IBM or its suppliers or licensors, or altering the terms and conditions of the applicable license agreement governing the use of IBM software. References in these materials to IBM products, programs, or services do not imply that they will be available in all countries in which IBM operates. Product release dates and/or capabilities referenced in these materials may change at any time at IBM's sole discretion based on market opportunities or other factors, and are not intended to be a commitment to future product or feature availability in any way. IBM, the IBM logo, Rational, the Rational logo, Telelogic, the Telelogic logo, and other IBM products and services are trademarks of the International Business Machines Corporation, in the United States, other countries or both. Other company, product, or service names may be trademarks or service marks of others.