Java Persistence: The Problem Space

Customer
| id: int    |
| name: String |
| creditRating: int |

Java
- JPA

Relational
- CUST
  | ID | NAME | C_RATING |

JAXB: Java Architecture for XML Binding

XML
- <customer id="...">
  - <name>...
  - ...
- </customer>

JSON
- { id="...",
  - name ="...",
  - ...
}

JPA: Java Persistence API

“JSON-B”
Software Evolution

• Computing architecture is constantly evolving: Mainframe, client/server, web/thin client, mobile/apps, ...

• Current technologies with increasing adoption include:
  – Cloud computing
  – HTML 5
  – NoSQL databases

• Java EE 7 is evolving to address many of these new requirements

• EclipseLink JPA and JAXB are also evolving!
New Features

• JSON Binding
• RESTful JPA – JPA RS
• Dynamic
  – Configuration
  – Provisioning
• NoSQL Database Support
• Tenant Isolation
JSON Binding / EclipseLink “JSON-B”

• Provides Java/JSON binding similar to EclipseLink JAXB’s Java/XML binding.

• Marshall Java domain model to and from JSON

• Currently no Java standard—EclipseLink interprets JAXB XML bindings for JSON

• Content-type selectable by setting property on Marshaller/Unmarshaller
XML and JSON from JAXB Mappings

```java
@XmlRootElement(namespace="urn:example")
public class Foo {

    @XmlAttribute
    private int id;

    @XmlElement(namespace="urn:example")
    private String bar;
}
```

---

XML

```xml
<foo xmlns="urn:example" id="123">
  <bar>Hello World</bar>
</foo>
```

---

JSON

```json
{"foo": {
  "id": 123,
  "bar": "Hello World"
}}
```
RESTful JPA

• Provides a service that exposes JPA mapped entities over REST via JAX-RS

• HTTP message body either XML or JSON

• Client
  – HTML 5 with JavaScript (primary focus)
  – JavaFX
JAX-RS with JPA—High Level Architecture

Client

Java EE Server

RDBMS

HTTP/S

JDBC
public class InvoiceService {

    public Invoice read(int id) {
        return null;
    }

    ...


@Stateless

public class InvoiceService {...

    public Invoice read(int id) {
        return entityManager.find(Invoice.class, id);
    }

    ...


JAX-RS with JPA Example – GET Invoice

@Path("/invoice")
@Stateless
public class InvoiceService {

    public Invoice read(int id) {
        return entityManager.find(Invoice.class, id);
    }

    ...

}
JAX-RS with JPA Example – GET Invoice

@Path("/invoice")
@Stateless
public class InvoiceService { ...

@GET
@Path("{id}")
public Invoice read(@PathParam("id") int id) {
    return entityManager.find(Invoice.class, id);
}
...

JAX-RS with JPA Example – GET Invoice

@Path("/invoice")
@Stateless

public class InvoiceService { ... 

@GET
@Path("{id}")
@Produces({"application/xml", "application/json"})

public Invoice read(@PathParam("id") int id) {
    return entityManager.find(Invoice.class, id);
}

...
JAX-RS with JPA Example – GET Invoice

@Path("/invoice")
@Stateless
public class InvoiceService {...

    @GET
    @Path("{id}")
    @Produces({"application/xml", "application/json"})
    public Invoice read(@PathParam("id") int id) {
        return entityManager.find(Invoice.class, id);
    }

...
JAX-RS with JPA

GET http://.../invoice/4
mapped to bean

Invoice Bean
Contract Bean
Payment Bean

Accounting PU
Contracting PU
Human Resources PU

Bean uses JPA

JAX-RS

JPA
JPA-RS

JAX-RS http://.../MyApp/Accounting/Invoice/... mapped to JPA-RS service

JPA-RS maps URI http://.../MyApp/Accounting/Invoice/... to **Accounting** PU and **Invoice** entity

GET http://.../MyApp/Accounting/Invoice/...
REST Resource Mapping

• REST requires URIs for identifiable resources
• Resources not 1:1 with classes
  – may be a graph of closely related objects
• Resources are connected via links
• Need a way to define Resource Model that can be leveraged by JAXB/JSON Binding
Resource Model

• Maps Java Object Model to REST Resources
EclipseLink Resource Model

• In development
• Resources (sub-graphs of domain graph) can be marshalled and unmarshalled (and reconnected)
• Links are being automatically generated
  – Currently requires use of JAXB annotations
• Future: simplify metadata declaration of resources
Summary

• Java is evolving—and EclipseLink is evolving too!
  – JSON Binding
  – JPA-RS
  – REST Resource Mapping
  – NoSQL
  – Tenant Isolation

• EclipseLink is the center of innovation in Java persistence