ORACLE

Future<JavaEE>

Jerome Dochez, GlassFish Architect

The following/preceding is intended to outline our general product direction. It is intended for information purposes only, and may not be incorporated into any contract. It is not a commitment to deliver any material, code, or functionality, and should not be relied upon in making purchasing decisions.

The development, release, and timing of any features or functionality described for Oracle's products remains at the sole discretion of Oracle.

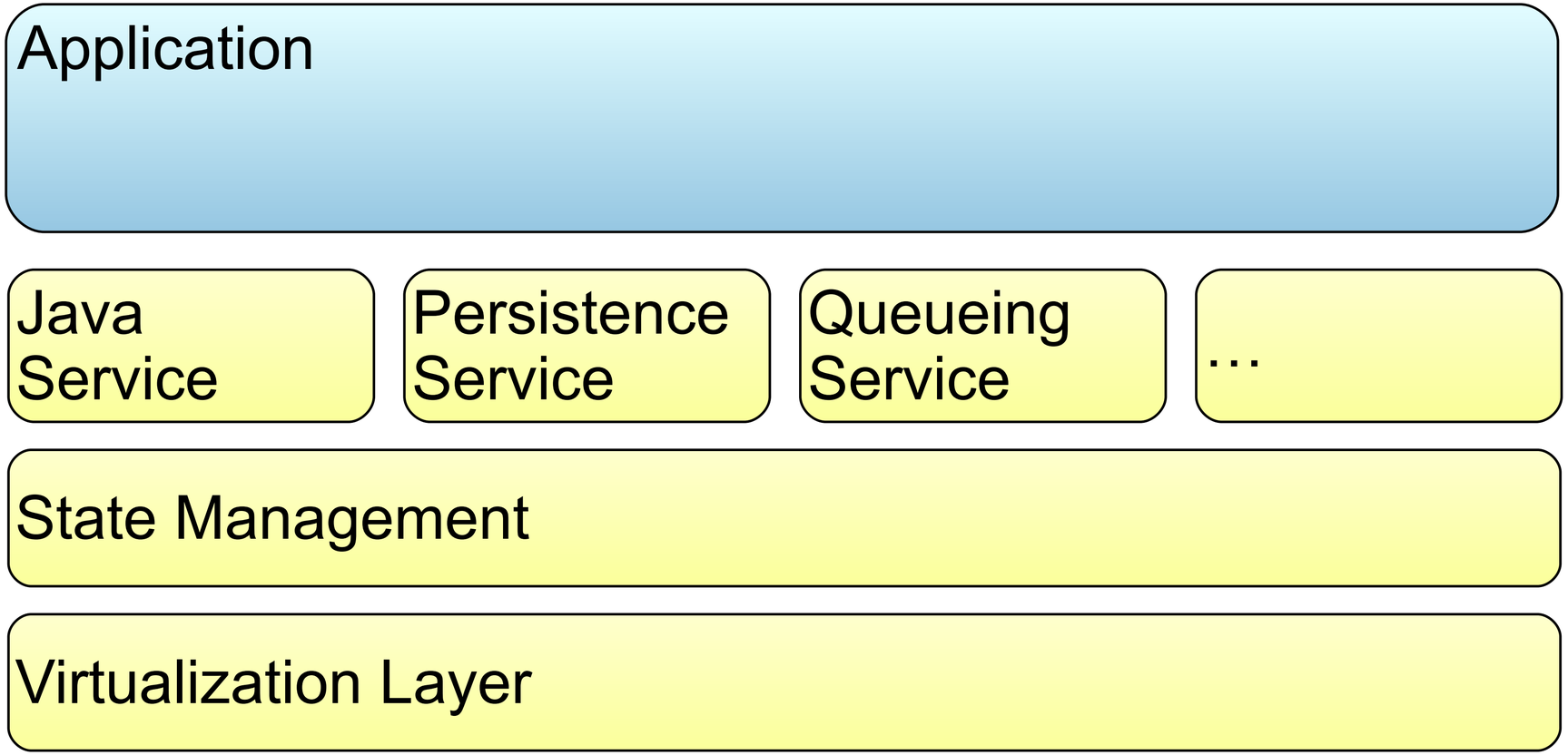
Java EE for the Cloud

- Tighter requirements for resource and state management
- Better isolation between applications
- Potential standard APIs for NRDBMS, caching, other
- Common management and monitoring interfaces
- Evolution, not revolution

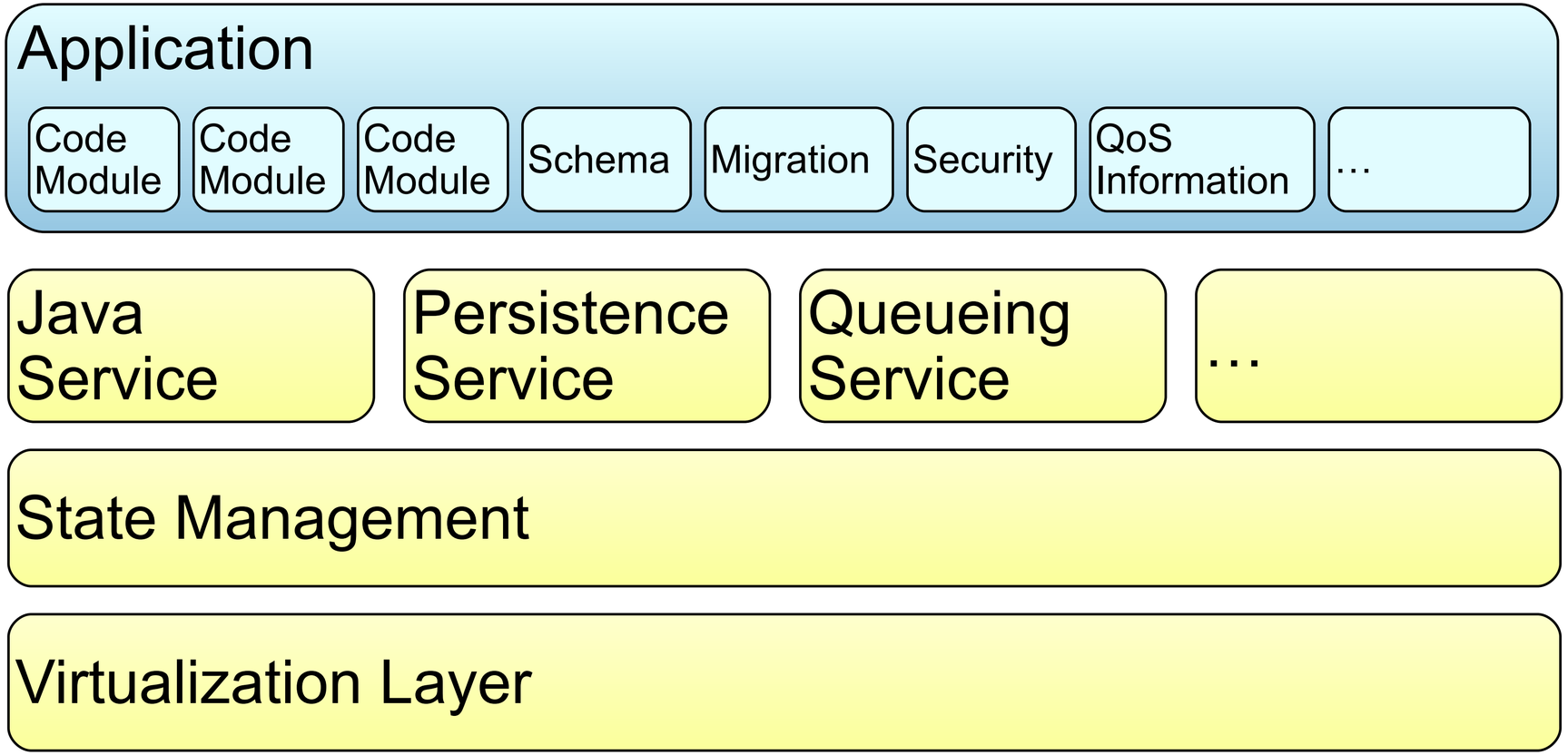
Better Packaging for the Cloud

- Apps are versioned
- Multiple versions can coexist
- Must deal with data versioning, upgrades, etc.
- Need ability to specify QoS properties
- Apps both expose and connect to services

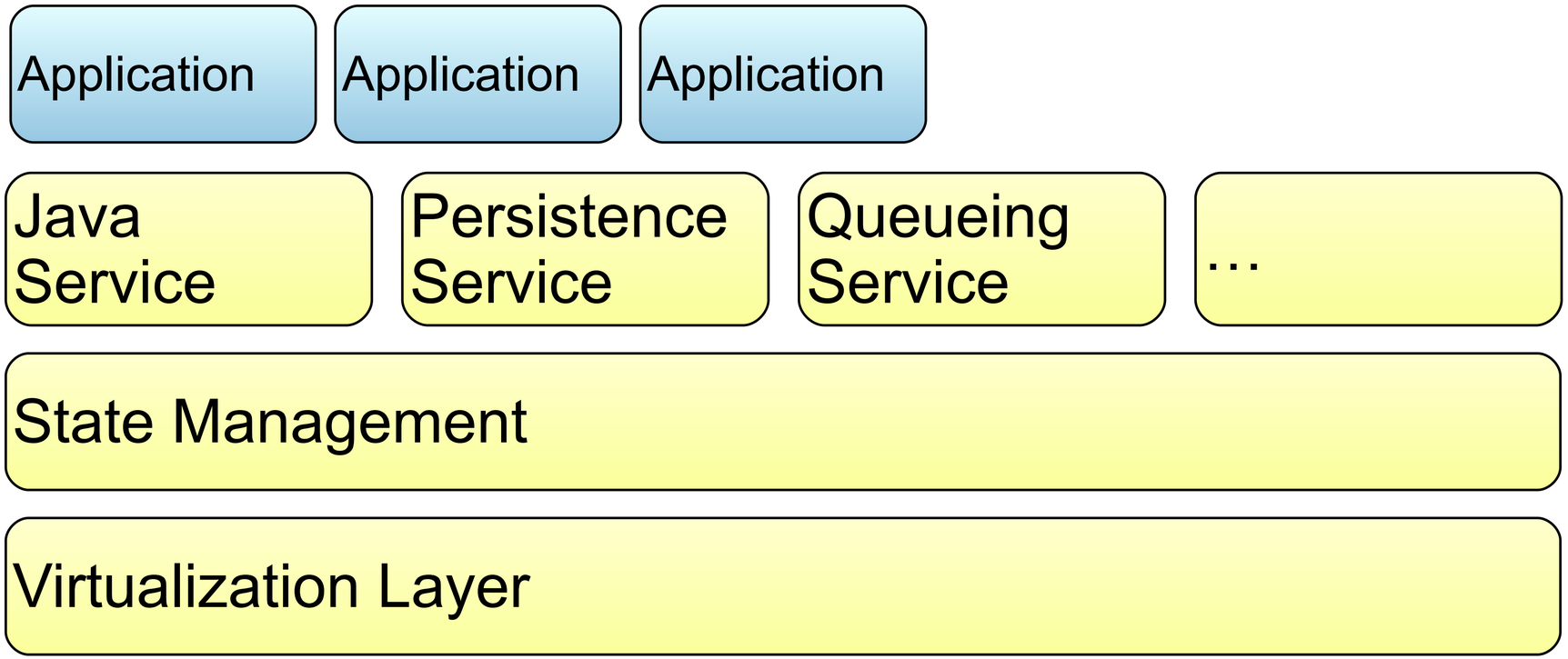
Cloud Platform



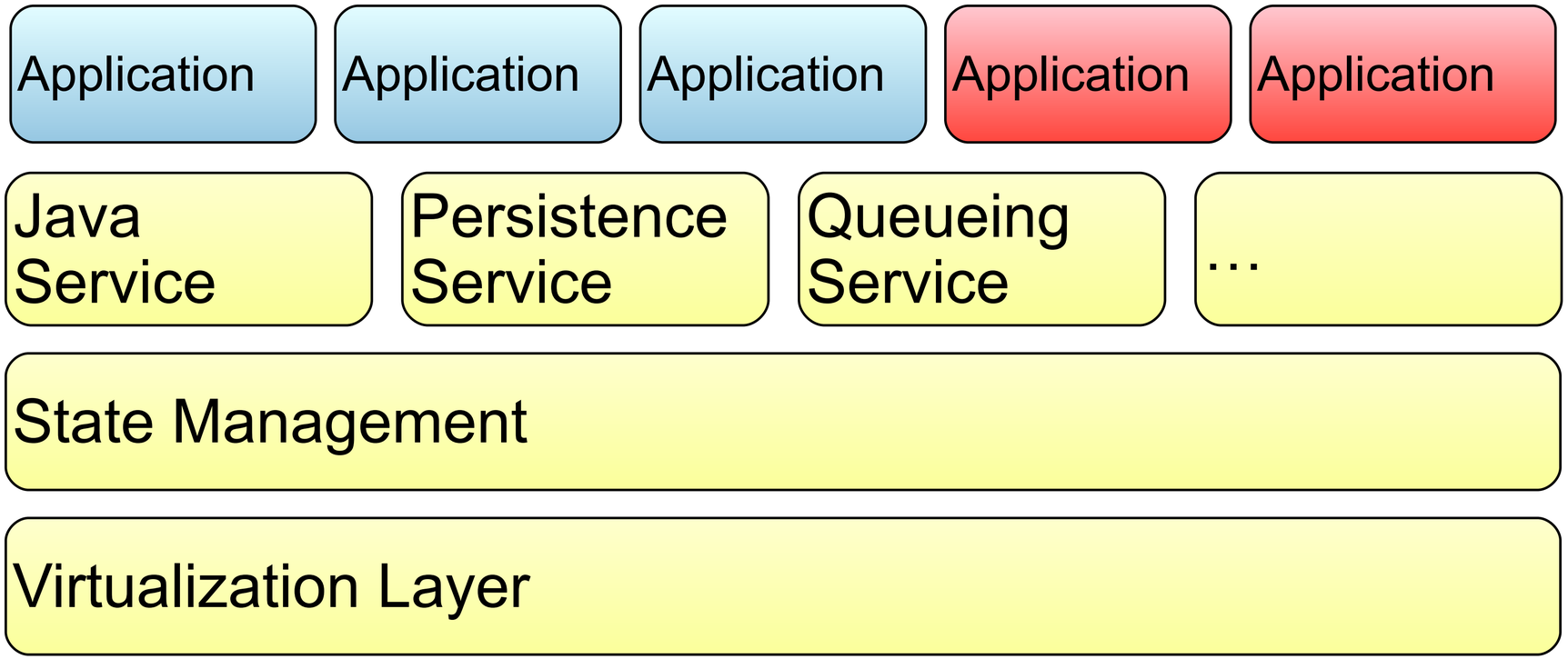
Cloud Platform



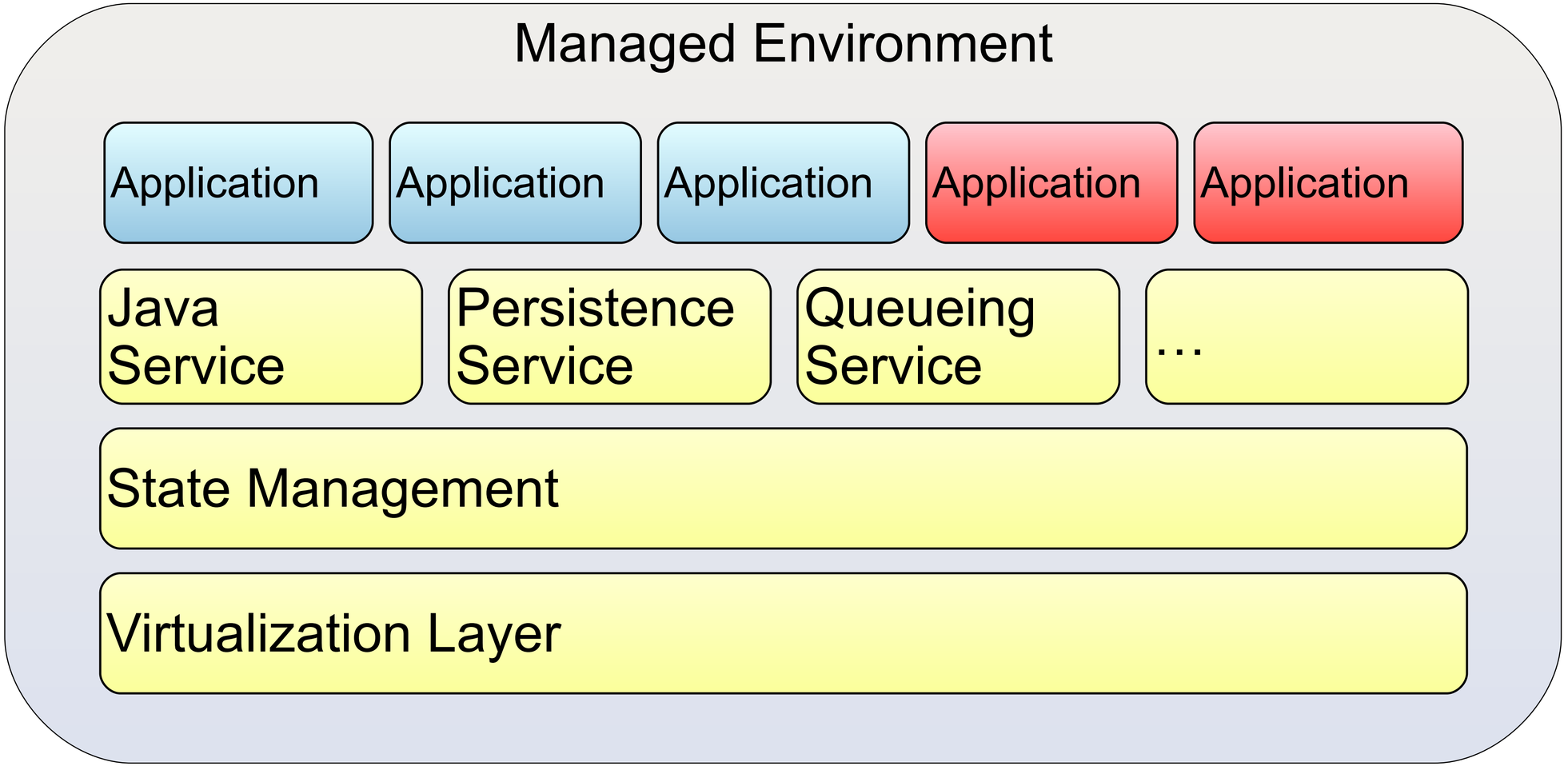
Cloud Platform



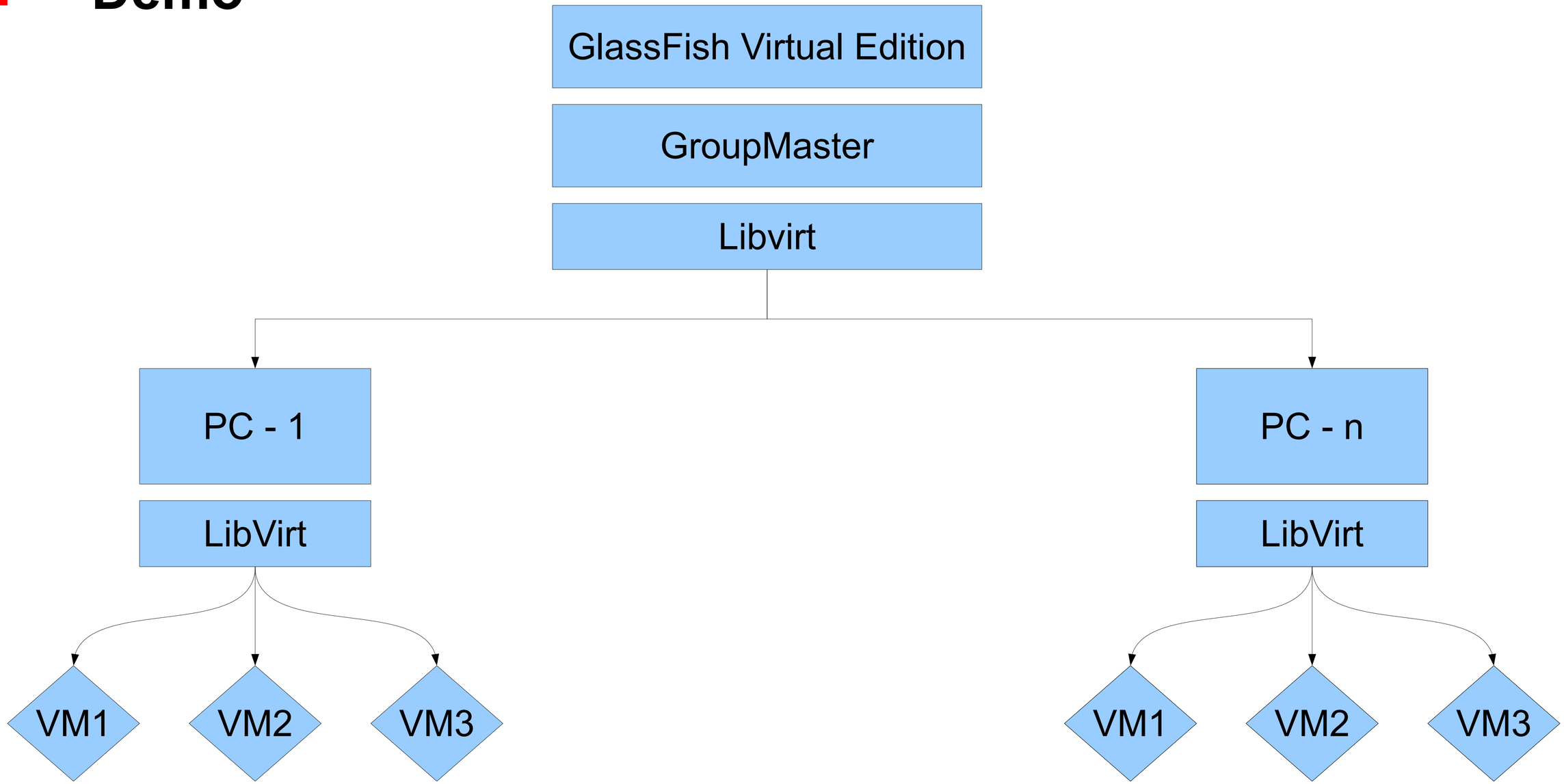
Cloud Platform



Cloud Platform



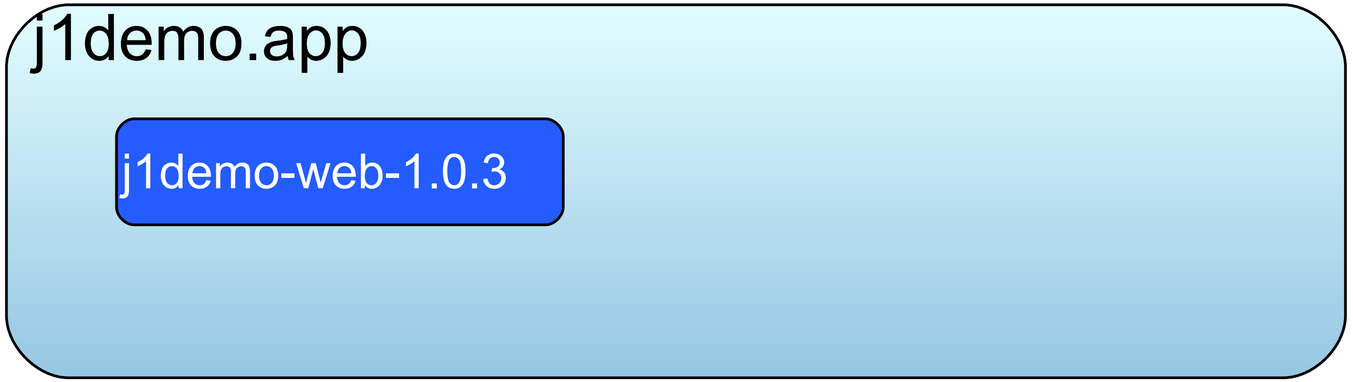
Demo


GlassFish Virtual Edition
GroupMaster
Libvirt
PC - 1
LibVirt
VM1
VM2
VM3
PC - n
LibVirt
VM1
VM2
VM3

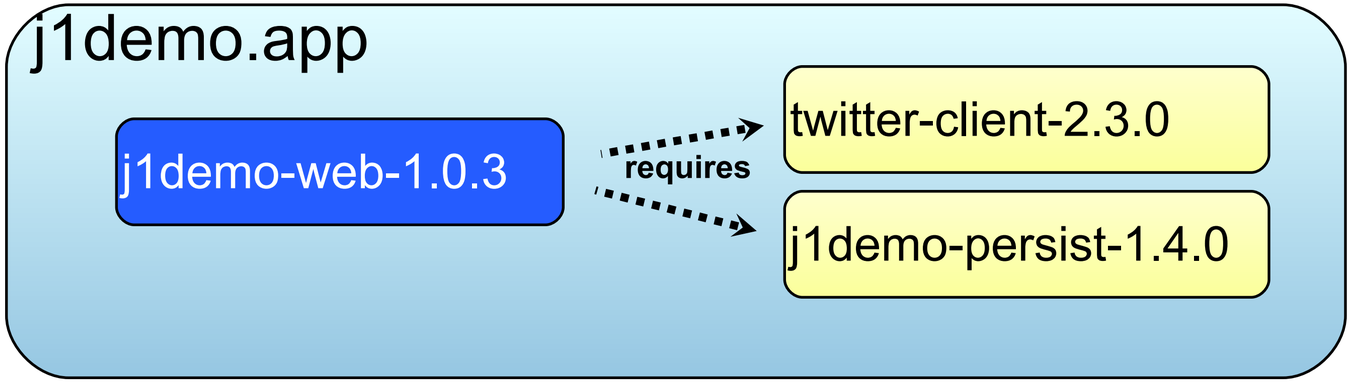
Modularity

- Build on Java SE 8 work
- Applications made of modules
- Dependencies are explicit
- Versioning is built-in
- Classloaders straightened out

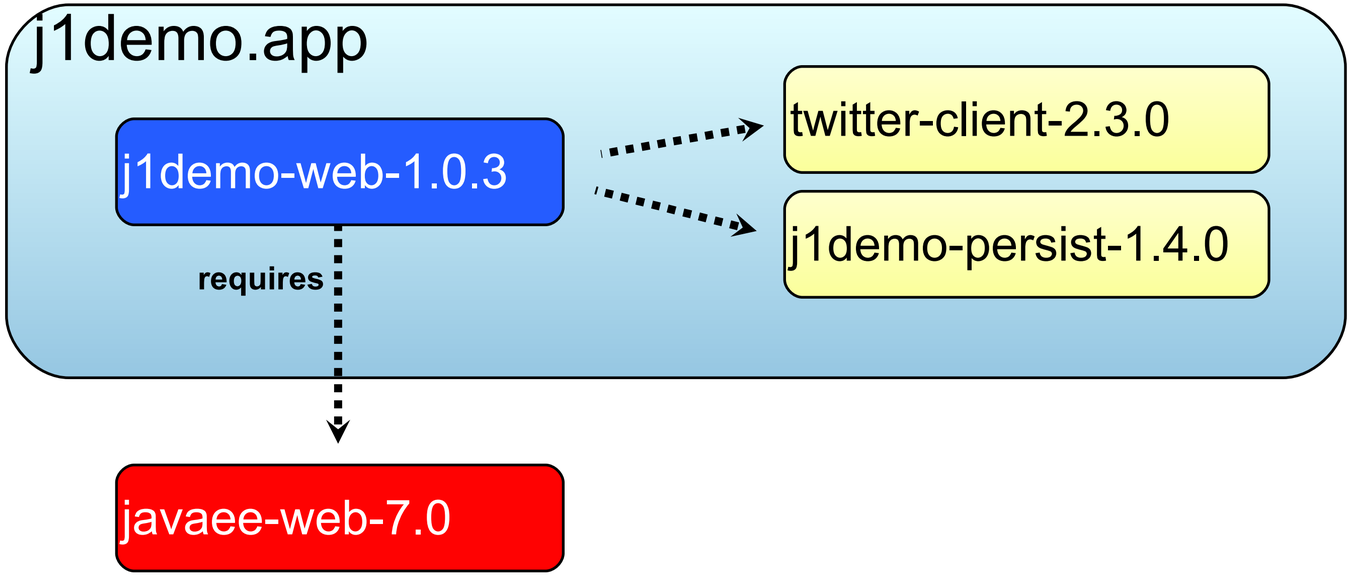
Modular Applications



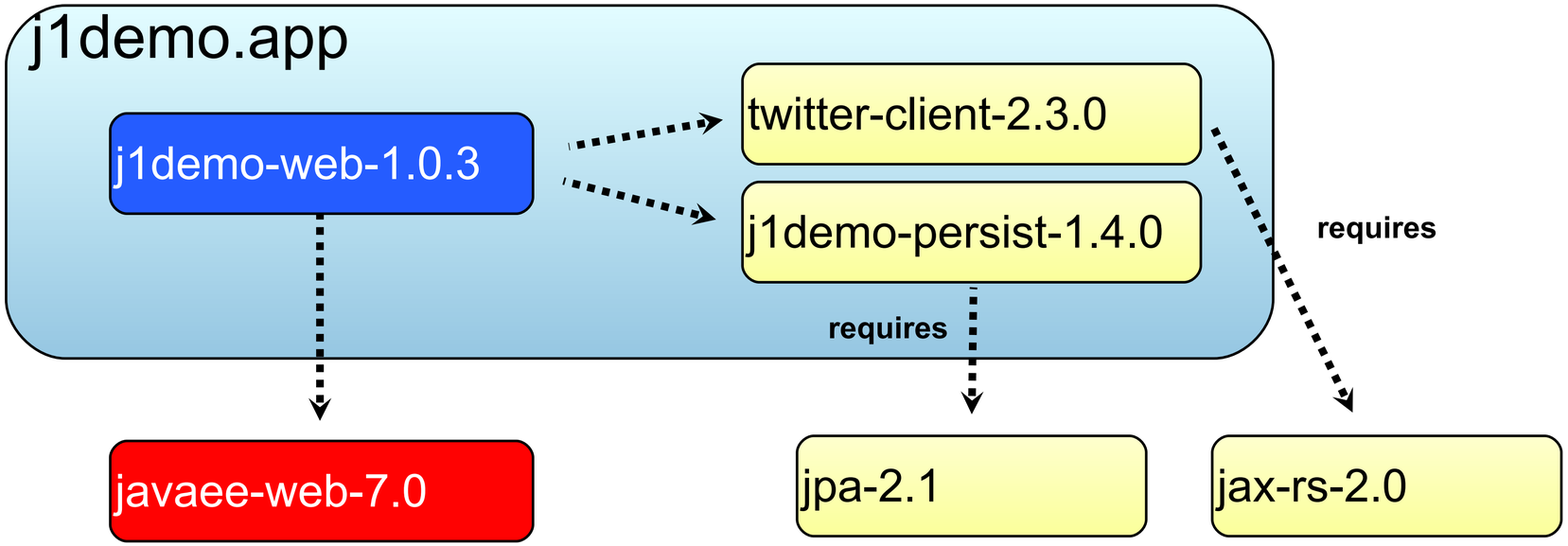
Modular Applications



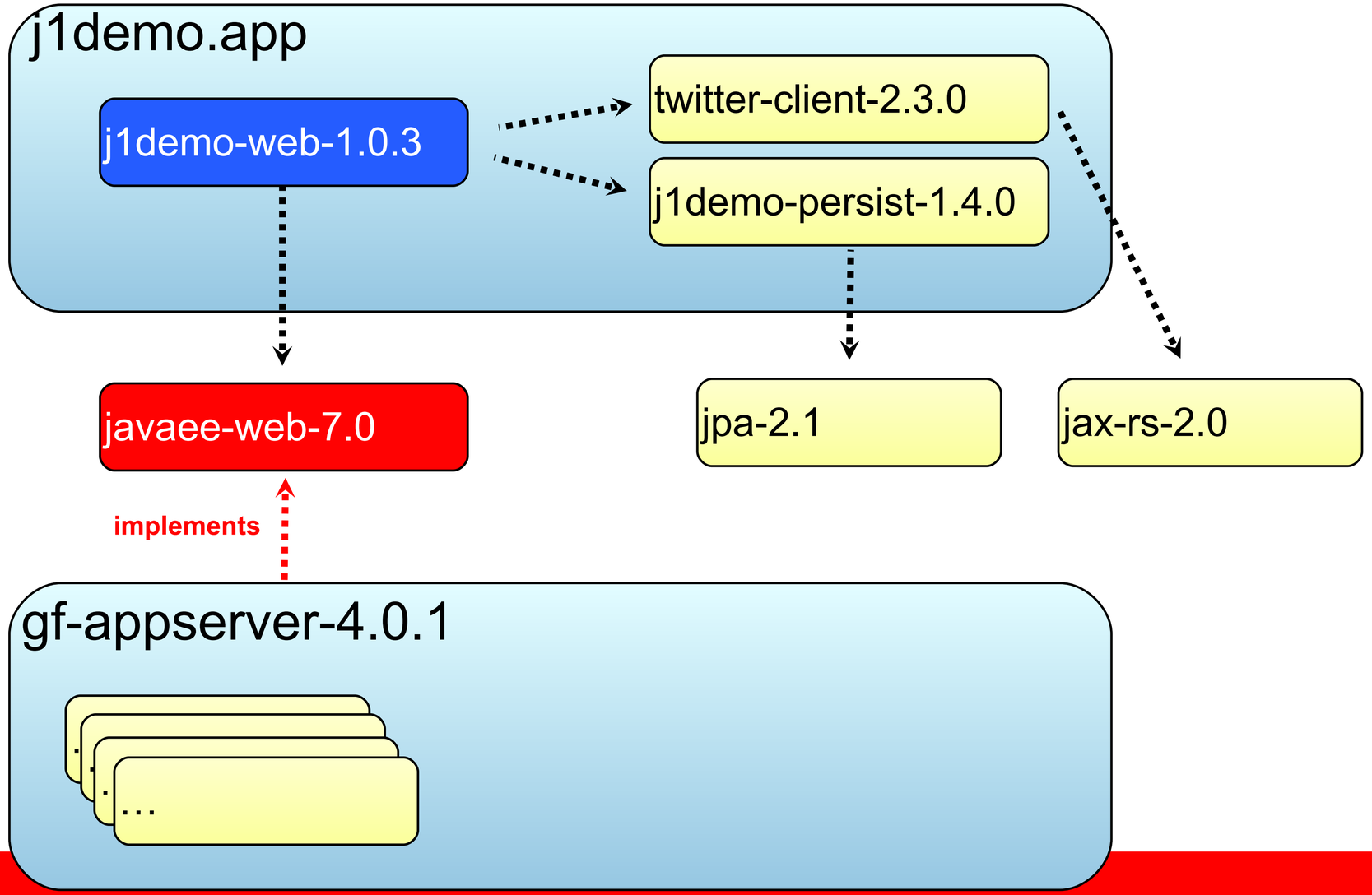
Modular Applications



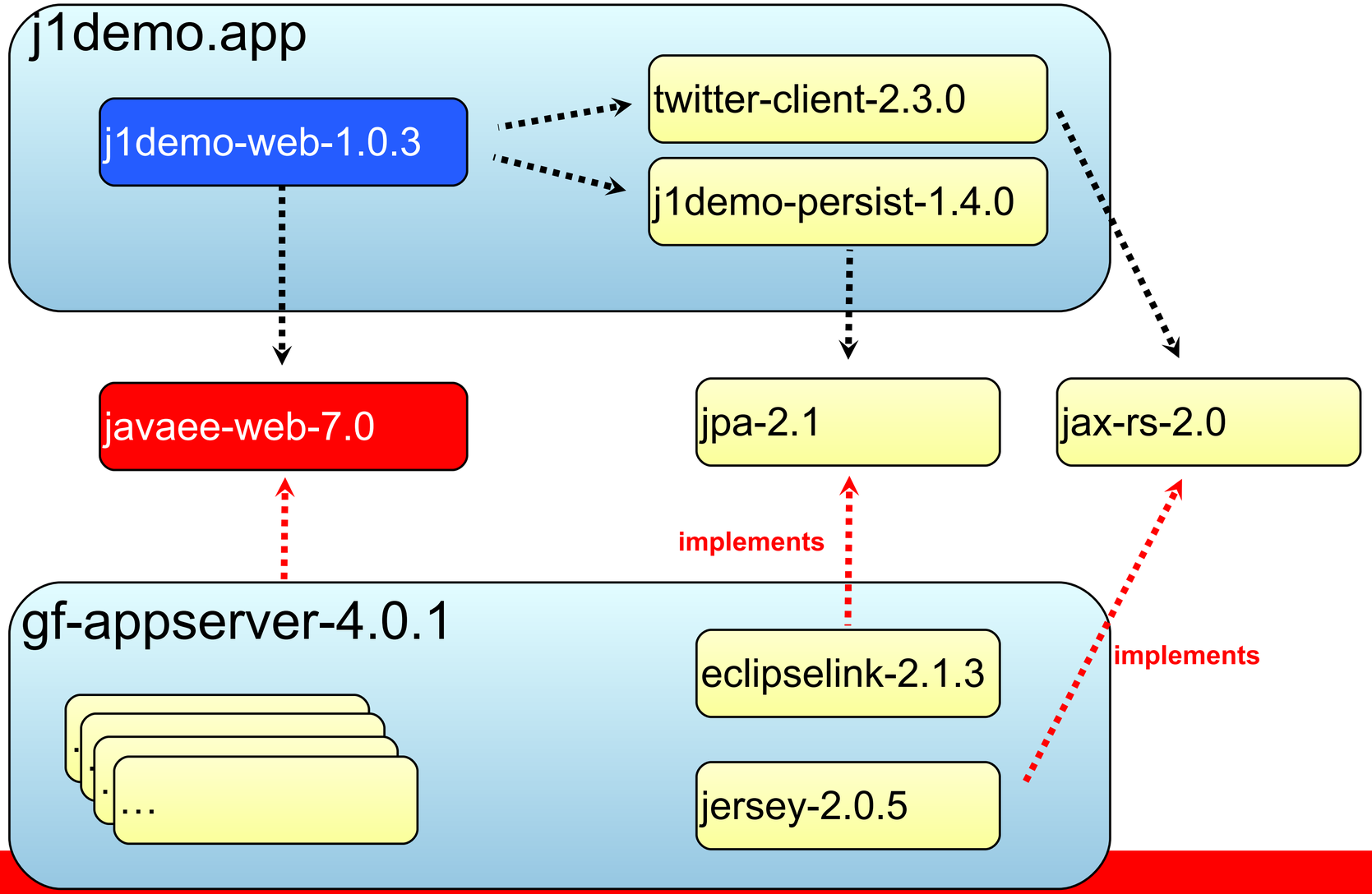
Modular Applications



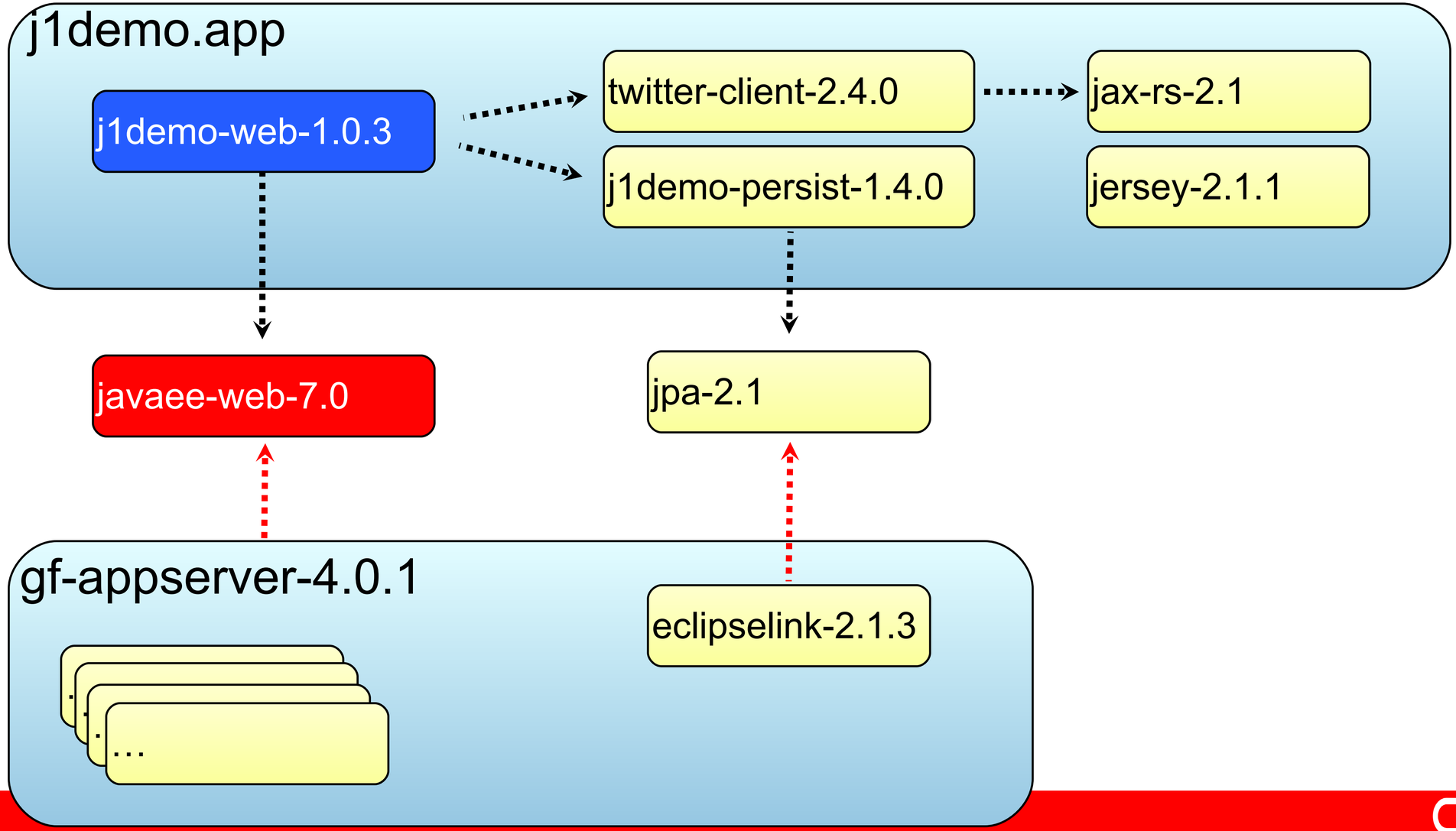
Modular Applications



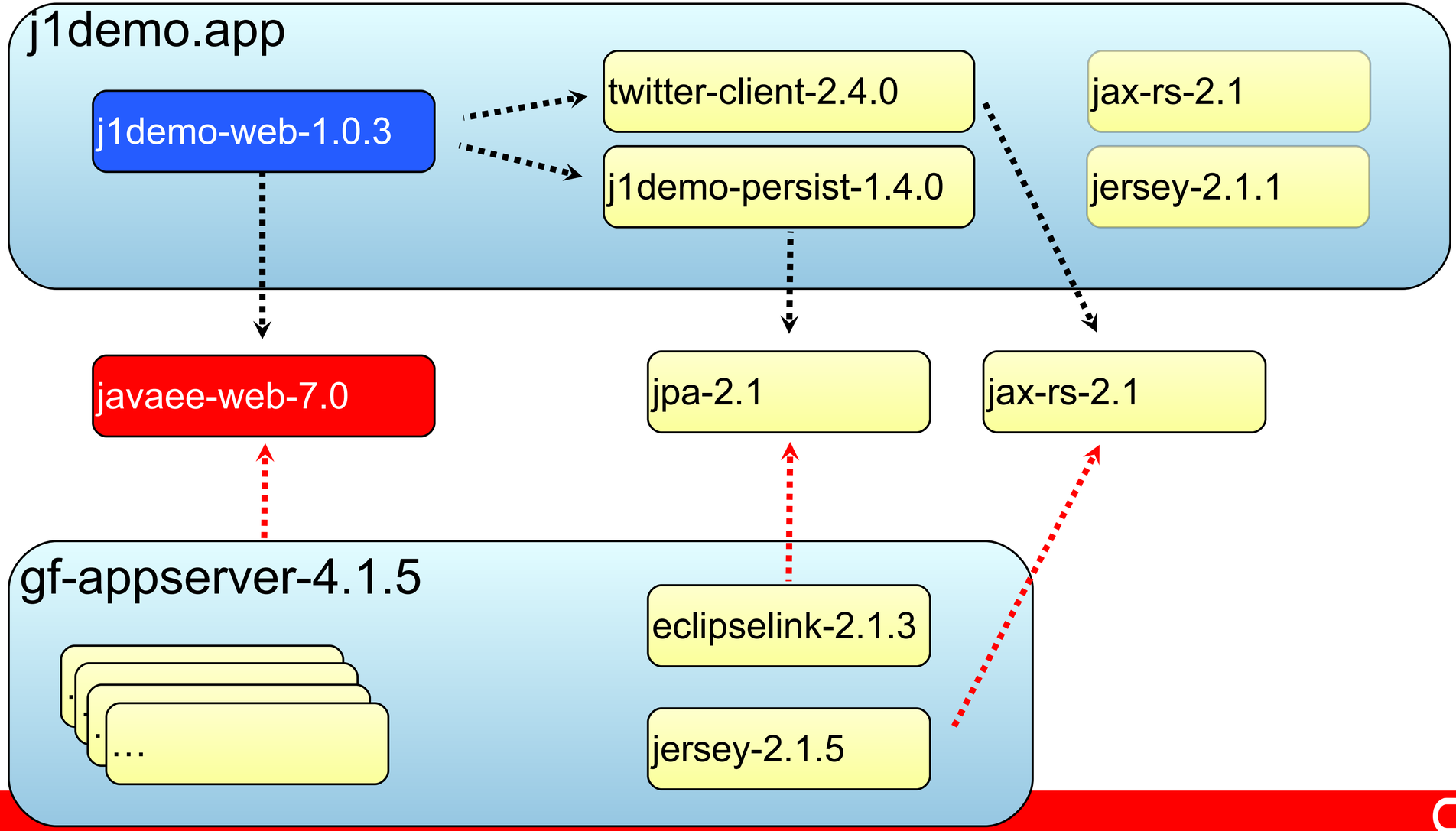
Modular Applications



Modular Applications



Modular Applications



Web Tier

- Web socket support
- Standard JSON API
- HTML5 support
- NIO.2-based web container

Web Socket Sample (w/Grizzly WebSocket API)

```
public class ChatApplication
        extends WebSocketApplication<ChatWebSocket> {

  protected ChatWebSocket createWebSocket(Connection c,
                                          ServerWebSocketMeta m)
    { return new ChatWebSocket(c, m, this); }

  public void onMessage(ChatWebSocket s, DataFrame frame) {
    String msg = frame.getAsText();
    s.sendJson(s.getUser(), msg);
  }

  public void onClose(ChatWebSocket s)
    { s.sendJson("system", s.getUser() + " left the chat"); }
}
```

Async Web Sample (w/Atmosphere)

```
@Path("/{topic}")
@Produces("text/plain;charset=ISO-8859-1")
public class PubSub {
    private @PathParam("topic") Broadcaster topic;

    @GET @Suspend
    public Broadcastable subscribe() {
        return new Broadcastable("",topic); }

    @POST @Broadcast
    @Consumes(MediaType.APPLICATION_FORM_URLENCODED)
    public Broadcastable publish(
                    @FormParam("message") String message) {
         return new Broadcastable(message,topic);
    }
}
```

JAX-RS 2.0 Client API (w/Jersey)

```
WebResource r = Client.create().resource("http://example.com");
WebResource content = r.path("containers").path("quotes");
content.path("1")
       .type(MediaType.TEXT_PLAIN)
       .put("Something is rotten in the state of Denmark");

ClientResponse response = content.path("1")
                  .header("If-Modified-Since", lastModified)
                  .header("If-None-Match", etag)
                  .get(ClientResponse.class);

assertEquals(Response.Status.NOT_MODIFIED,
             response.getResponseStatus());
```

JSF 2.2

- HTML5 support
  - Semantic tags, browser feature detection
  - Use of HTML5 by components
  - Audio and video
- JSR-276 (tool support)
- Mobile renderkits
- Page resolver and application context manager API
- Performance

JMS 2.0

- Much needed update!
- Ease of development makeover
- Better integration with application servers
- Standardize some common vendor extensions

JPA 2.1 Candidate Features

- Converters / custom types
- “Fetch” plans / profiles
- Additional mapping metadata
- User-defined naming strategies
- More flexibility in use of generated values
- Immutable attributes; readonly entities
- More flexible XML descriptors
- Additional event listeners and callbacks
- Improved control of persistence context synchronization
- Support for dynamic definition of persistence unit
- Extension of Metamodel API to ORM
- Methods for dirty detection
- Additional unwrap methods
- Support for stored procedures
- Additional built-in functions
- Database and vendor function invocation support
- Downcasting
- Outer joins with ON conditions
- Update and delete criteria queries
- Mapping between JPQL and criteria queries
- Improved result type mapping for native queries

Java EE 7 Platform Roadmap

- JAX-RS 2.0 and JPA 2.1 filed few months back
- Java EE 7, EJB 3.2 filed last week.
- Target completion in 2012

Java EE 6 Platform

Available Today

http://www.oracle.com/javaee