Is Enterprise Java Still Relevant?

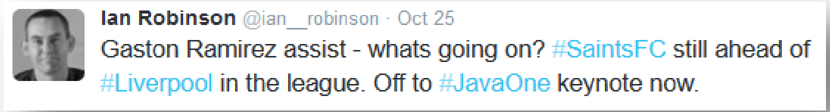
Ian Robinson, IBM Distinguished Engineer, WebSphere Foundation Chief Architect
@ian__robinson

Erin Schnabel, IBM WebSphere developer/engineer/guru/evangelist
@ebullientworks

About Me

- IBM Distinguished Engineer
- WebSphere Foundation Chief Architect
- 25 years experience in transaction processing and distributed enterprise computing
- WebSphere product strategy & development and enterprise Java standards
  - Including spec lead for a JSR with only 2 digits in it.
- Travels a lot, based in IBM's Hursley lab in the UK (near Southampton).
- Season ticket holder for one of the least fashionable clubs in English football:

Ian Robinson


Ian Robinson @ian__robinson · Oct 25
Gaston Ramirez assist - whats going on? #SaintsFC still ahead of #Liverpool in the league. Off to #JavaOne keynote now.

Platform Boundaries Are Breaking Down In Cloud…

…And Being Surrounded By Containers

Java EE – Relevant or Elephant?

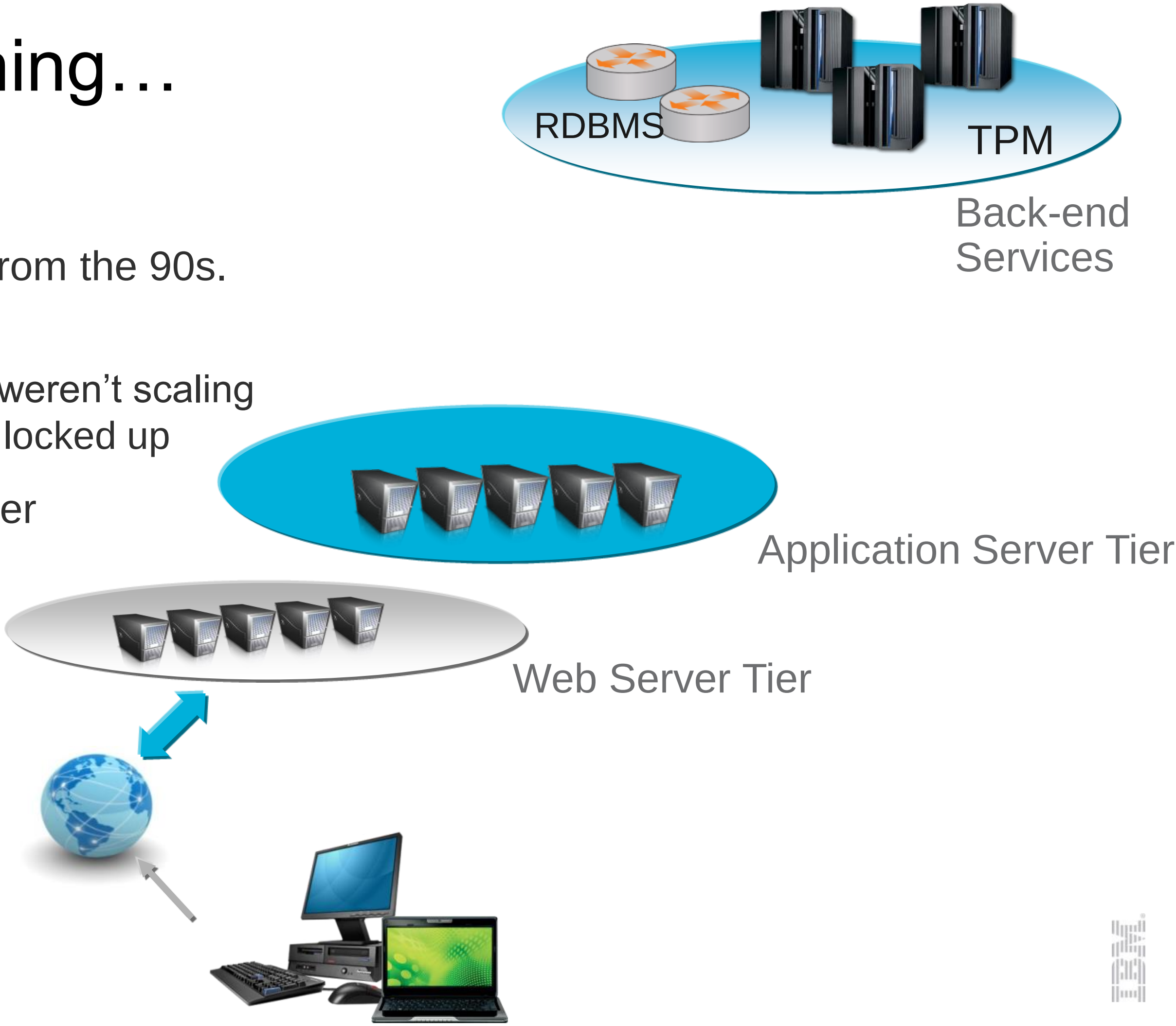
http://parentrap.org/2014/09/29/easy-parenting-part-iii-the-donts-list/

In The Beginning…

- Was e-commerce

← Genuine IBM template from the 90s.

- But
  - Web servers and CGI weren't scaling
  - The enterprise was all locked up
- The middle tier App Server emerged to scale up the front end and reduce load on the back-end.
- Application architecture still up for grabs.

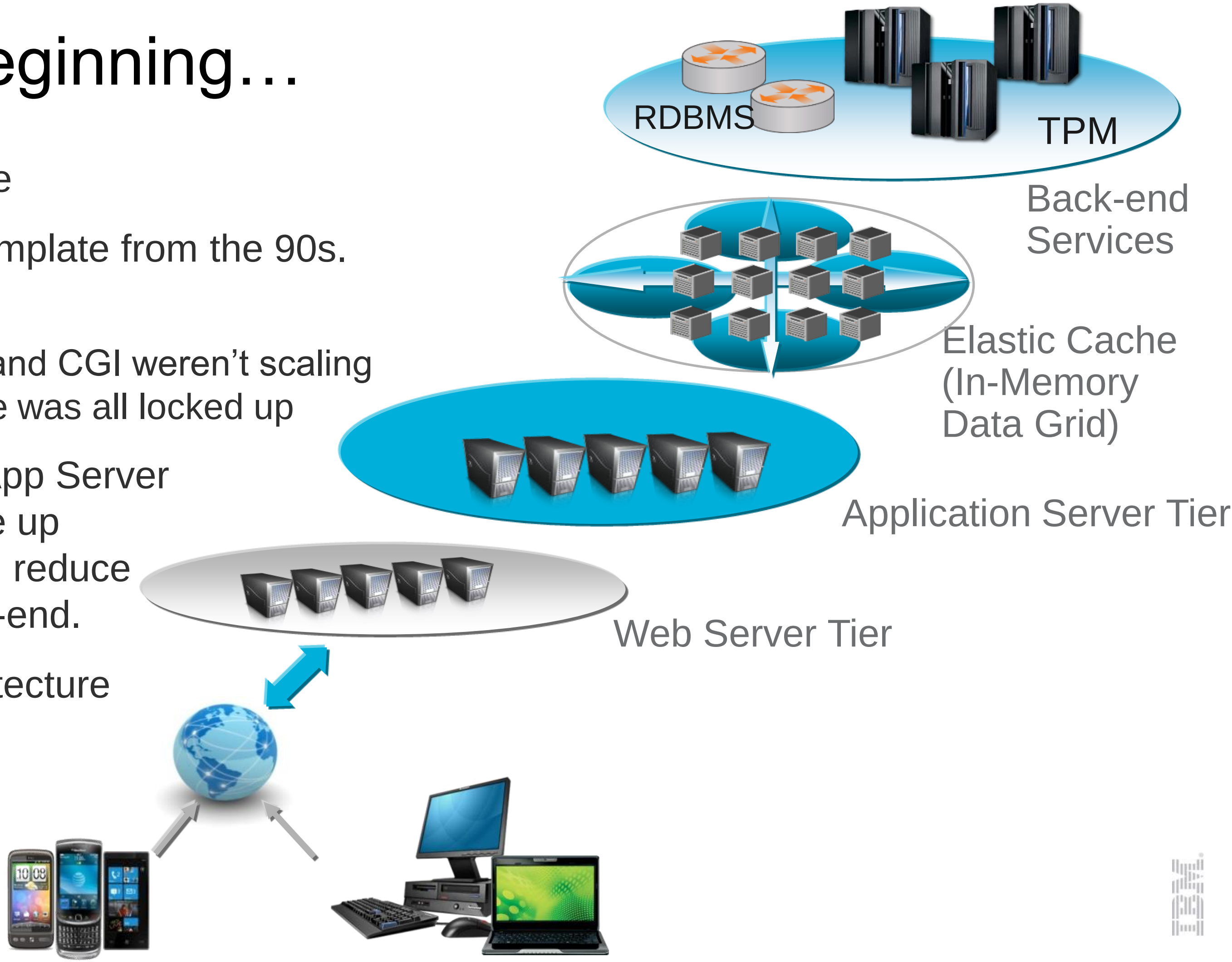


In The Beginning…

- Was e-commerce

← Genuine IBM template from the 90s.

- But
  - Web servers and CGI weren't scaling
  - The enterprise was all locked up
- The middle tier App Server emerged to scale up the front end and reduce load on the back-end.
- Application architecture still up for grabs.

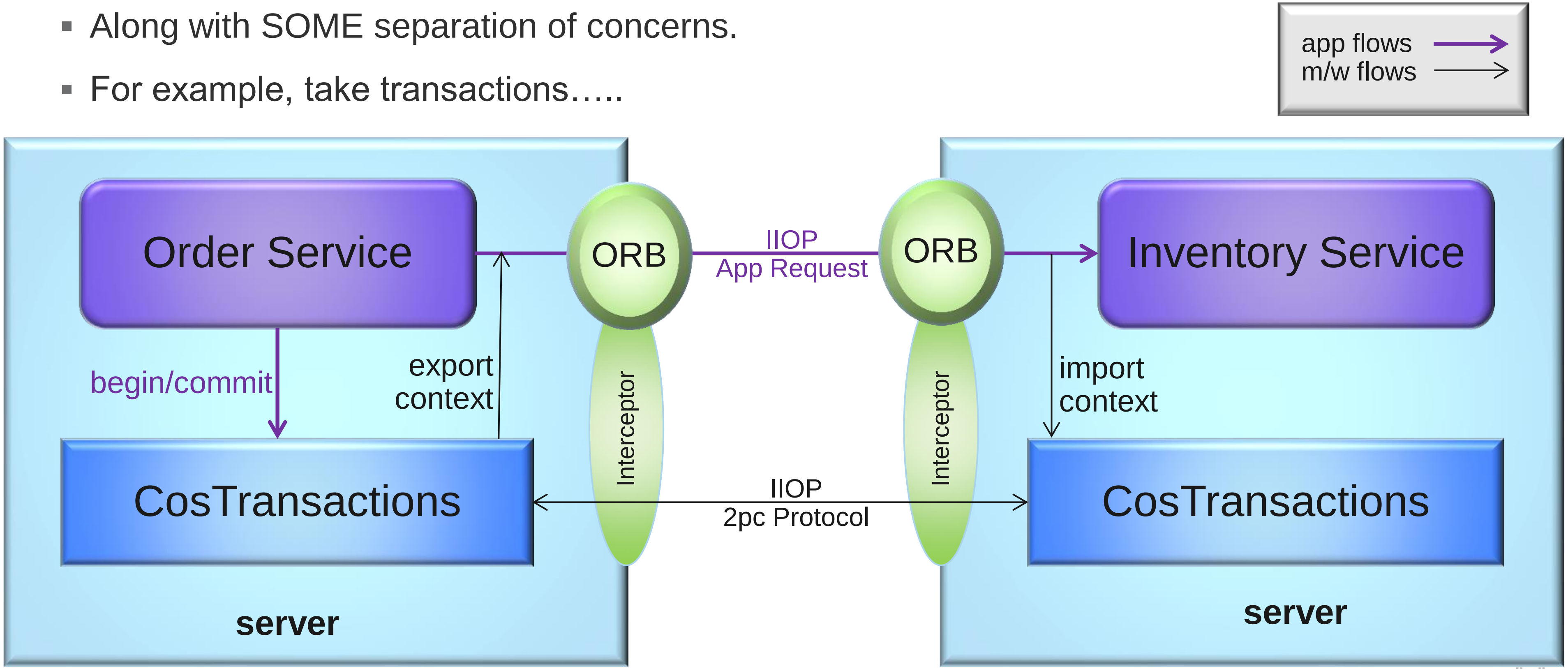


You’re Not On the List, You’re Not Coming In

- Servlets and JSPs were an entry point for the App-Server middle tier, but Java was not a complete enterprise architecture.
- For middle-tier enterprise computing, leave it to the Grown Ups….

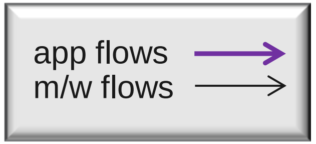
CORBA – Great for Middleware

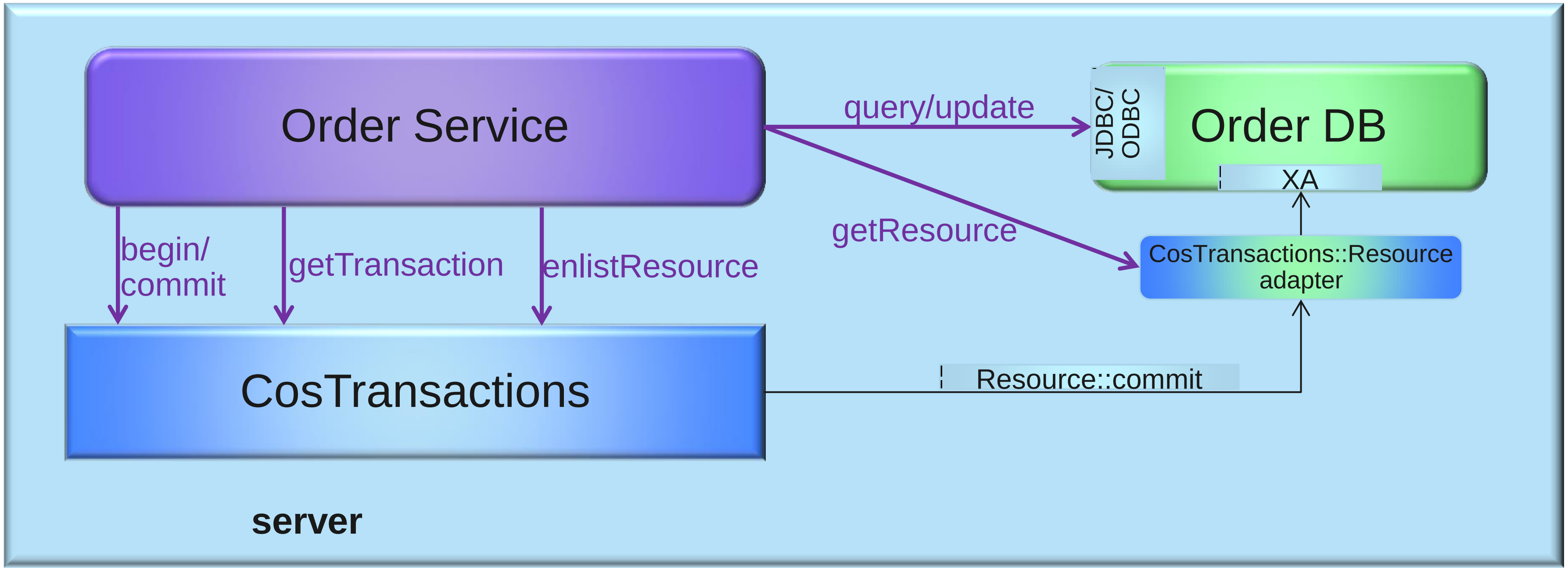
- CORBA gave us the underpinnings of distributed enterprise computing.
- Along with SOME separation of concerns.
- For example, take transactions…..



CORBA – Not So Great for Applications

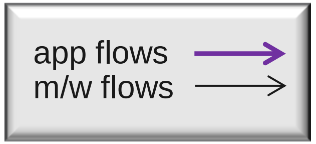
- No real separation between application and middleware
- No distinct application container contract
- Language agnosticism not so good for applications

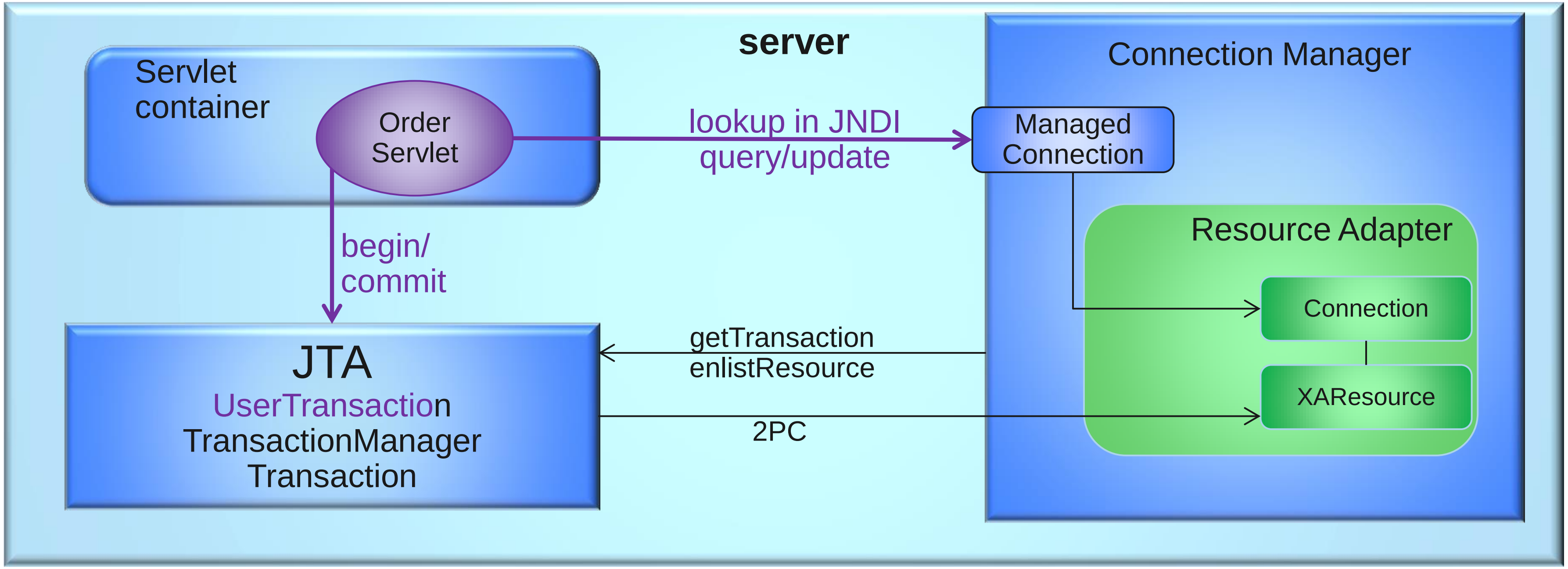




Enterprise Java Containers – Separating Concerns

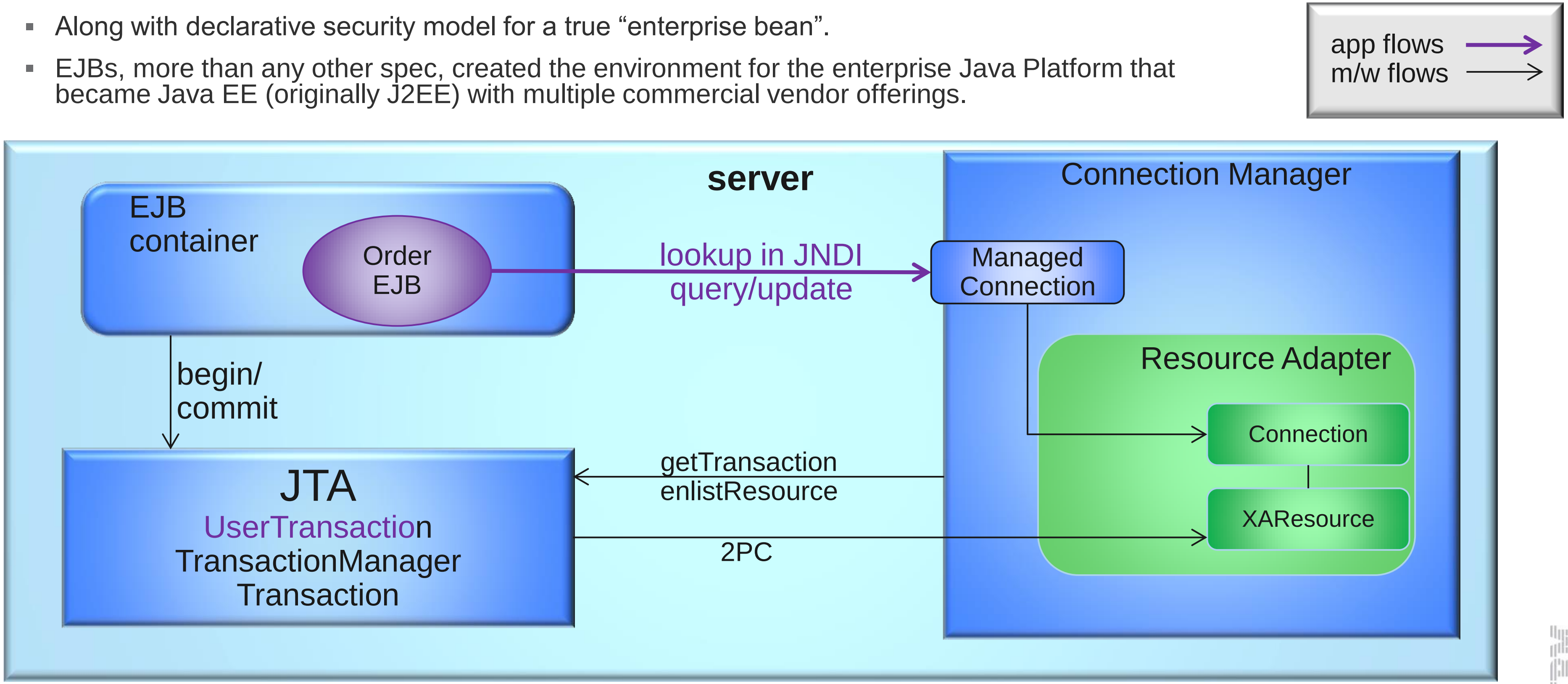
- Enterprise Java introduced application containers and defined proper separation of concerns between application components and the containers they run in.
- In the Transaction example, this was started with the JTA and JCA specs.
- The application is responsible ONLY for transaction demarcation.





The Managed Component – EJBs!

- EJBs came next for even greater simplification
- Example: for the first time, transactions can be completely declarative and decoupled from app logic
- Along with declarative security model for a true "enterprise bean".
- EJBs, more than any other spec, created the environment for the enterprise Java Platform that became Java EE (originally J2EE) with multiple commercial vendor offerings.



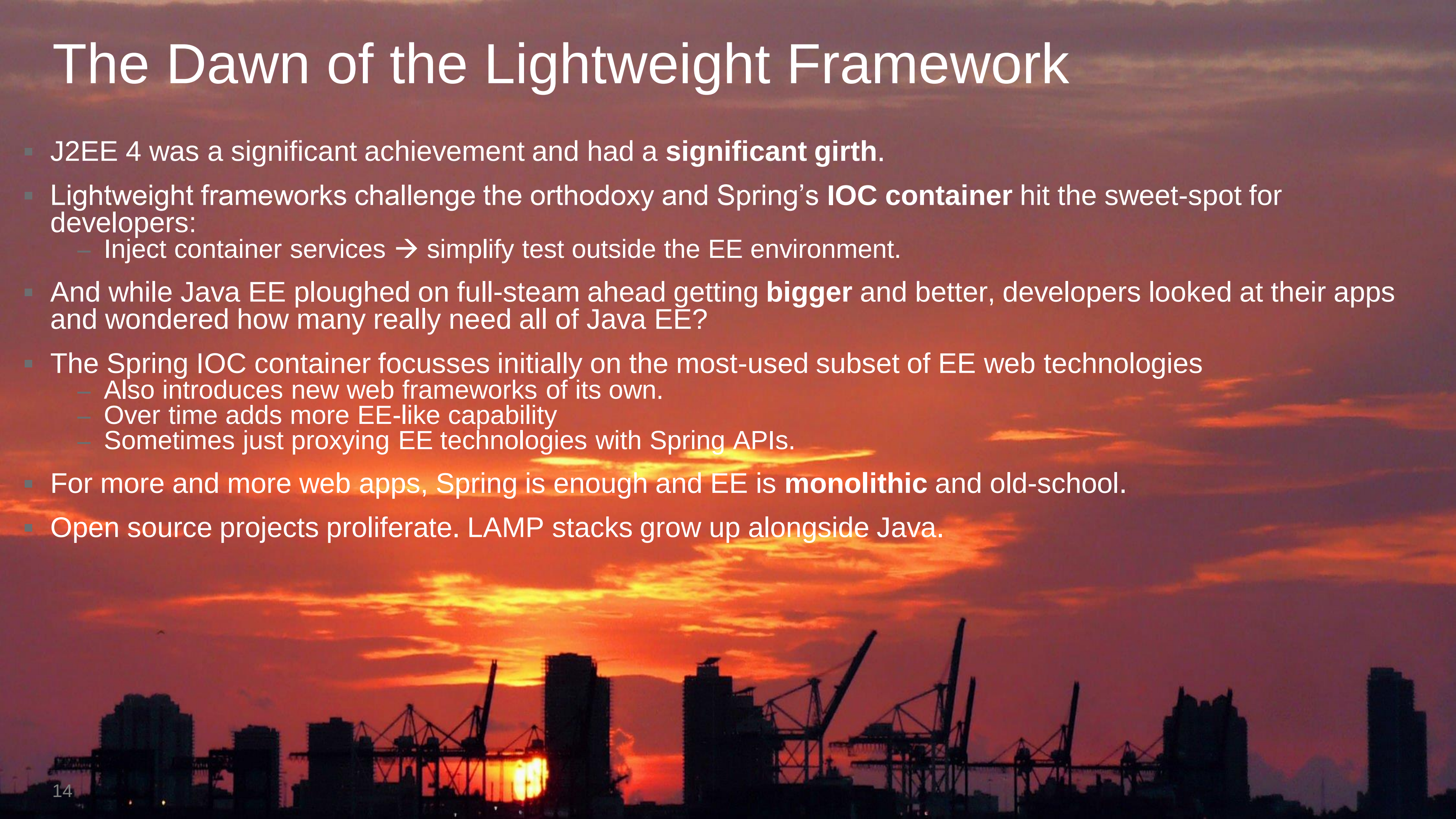
In the Garden, the Flowers and the Weeds Grew

- EE specifications evolved
- New ones were added as SOAP-based web services came into vogue
- EE got bigger and better.
- And bigger and bigger.

www.flickr.com/photos/ervins_strauhmanis

The Dawn of the Lightweight Framework

- J2EE 4 was a significant achievement and had a **significant girth**.
- Lightweight frameworks challenge the orthodoxy and Spring's **IOC container** hit the sweet-spot for developers:
  - Inject container services → simplify test outside the EE environment.
- And while Java EE ploughed on full-steam ahead getting **bigger** and better, developers looked at their apps and wondered how many really need all of Java EE?
- The Spring IOC container focusses initially on the most-used subset of EE web technologies
  - Also introduces new web frameworks of its own.
  - Over time adds more EE-like capability
  - Sometimes just proxying EE technologies with Spring APIs.
- For more and more web apps, Spring is enough and EE is **monolithic** and old-school.
- Open source projects proliferate. LAMP stacks grow up alongside Java.



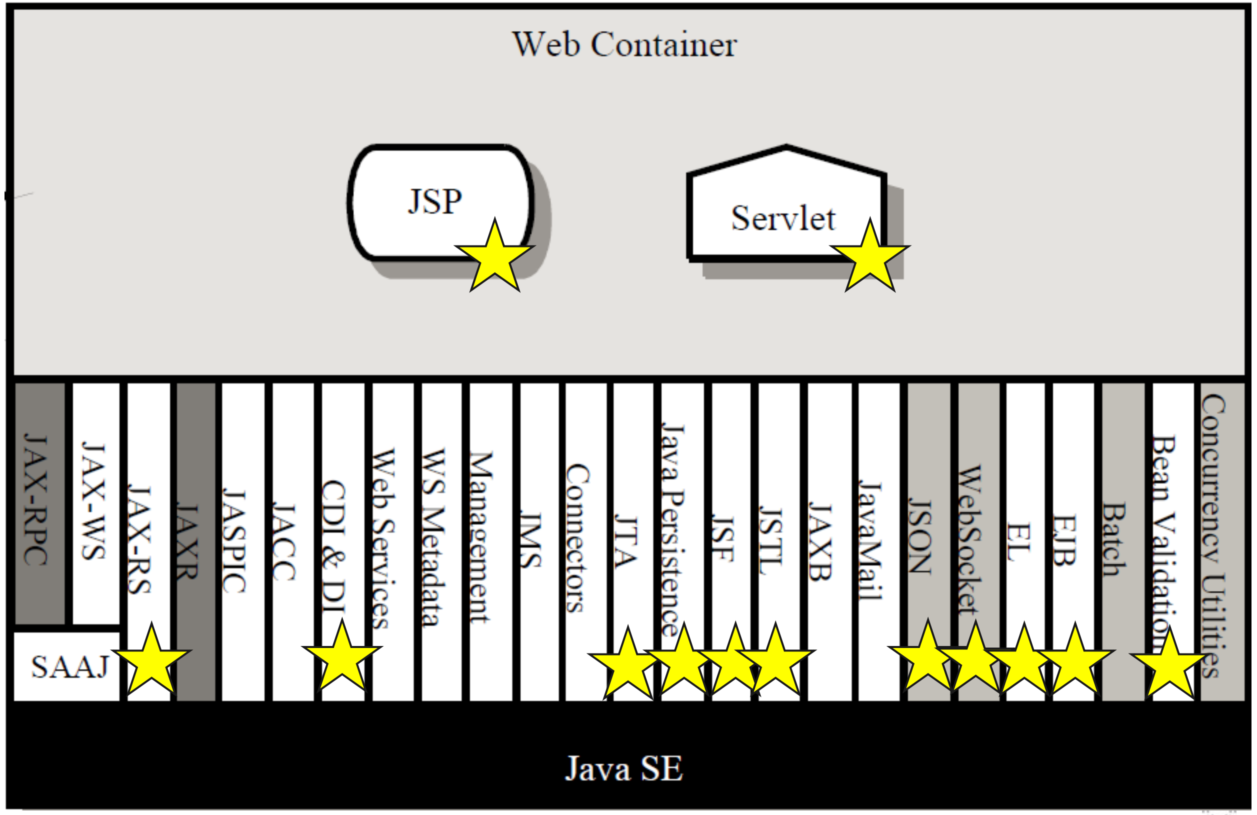
Java EE Simplification

- The Java EE Platform has **strict compatibility** rules
  - Spring and arbitrary collections of frameworks have none of that and have the flexibility to be as large or small as they like.
- With Java EE, you know how everything is supposed to behave with everything else and there are strict rules to demonstrate compliance.
  - Everything has to be there.
  - Difficult to look agile against less, *if less is enough*.
- EE5 embraced Spring-like IoC for **EJB3** and EE6 introduced **CDI** components to enable resources to be contextualized to web-request scope.
- EE6 introduced a sub-set **Web Profile** with separate compliance against a smaller set of coherent technologies.

Is Web Profile Enough?

- ***More*** than enough for many apps
- Nowhere near granular enough for next generation applications.

★ Web Profile specification



Modular Java EE - Does OSGi Help Here?

EEG - Enterprise Expert Group

- Does OSGi help Enterprise Java?
- OSGi is a **mature** and **well-used** Java Modularity System.
- OSGi Enterprise Spec added to OSGi R5 following creation of OSGi Enterprise Expert Group to bring enterprise application technologies to OSGi.
- OSGi applications can run well in the EE environment
  - Variety of ways these can be deployed, as EE or non-EE apps.
- Various open source projects in Apache and Eclipse developed specifications for how enterprise OSGi applications could consume EE technologies like servlets, JNDI, JPA, transactions, JMX as well as Spring-like managed beans and web components.
- Major benefit of this work was the “feature” construct in OSGi R5.

Apache ARIES

Karaf

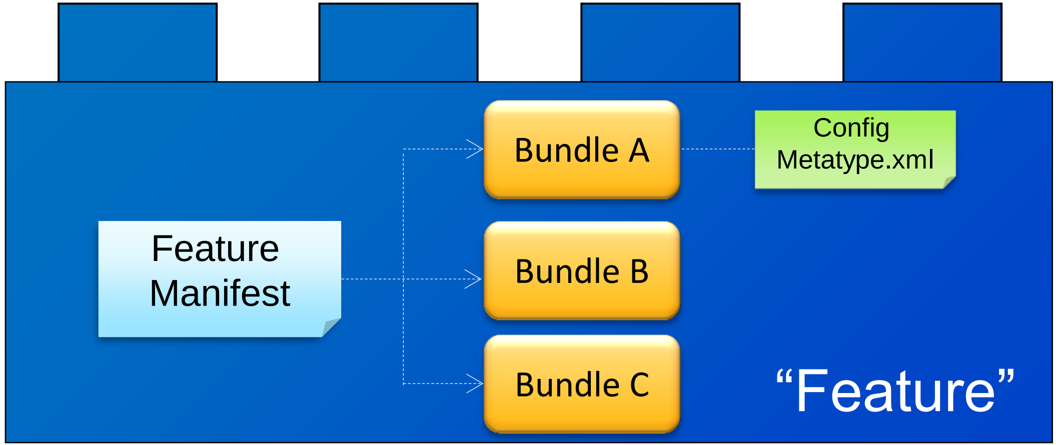
The WebSphere Omelette Challenge

- In WebSphere we primarily use OSGi to build an EE compliant but fully-modular application server for Java EE applications.
  - For this, OSGi is all on the "inside" and not visible to Java EE applications which are deployed as standard EARs or WARs.
- We did this in response to the following challenge in 2011:
  - Create a lightweight profile of WebSphere AppServer that starts in under 2 seconds
  - Make it completely dynamic for all changes to configuration
  - Provide an unzip install ~60 Meg in size
  - Provide complete backward compatibility
  - Remain EE compliant
  - But **don't** break any eggs.

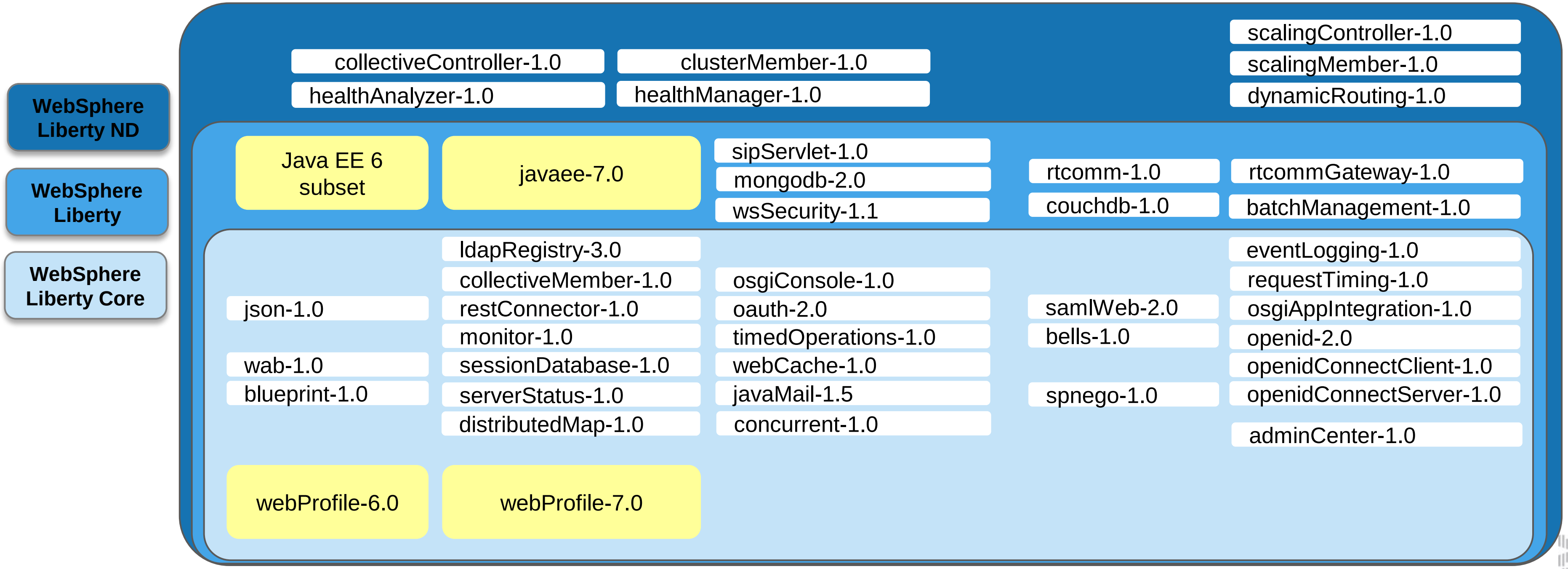
A La Carte Features, *Prix Fixe* EE Profiles

- We created WebSphere Liberty to supports arbitrary combinations of runtime “**features**” in addition to pre-defined sets for Web Profile and Full Java EE.
- Remember the eggs: any app running on WebSphere Liberty **runs unchanged** on the previous version of WebSphere.

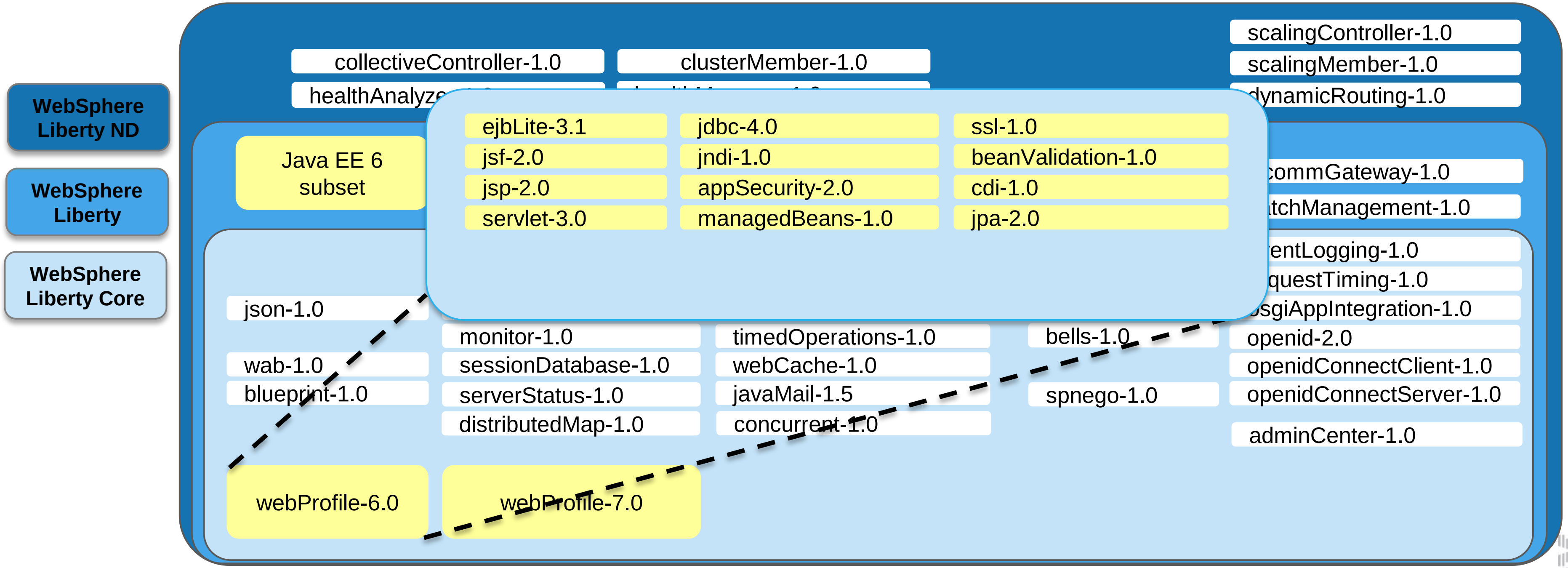
- Runtime bundles loaded and configured by OSGi subsystem-aware kernel as independent feature subsystems.
- Entirely self-contained metadata to describe bundle content, services published, & configuration metatypes.
- We use features as units of:
  - Deployment
  - Configuration
  - Extensibility



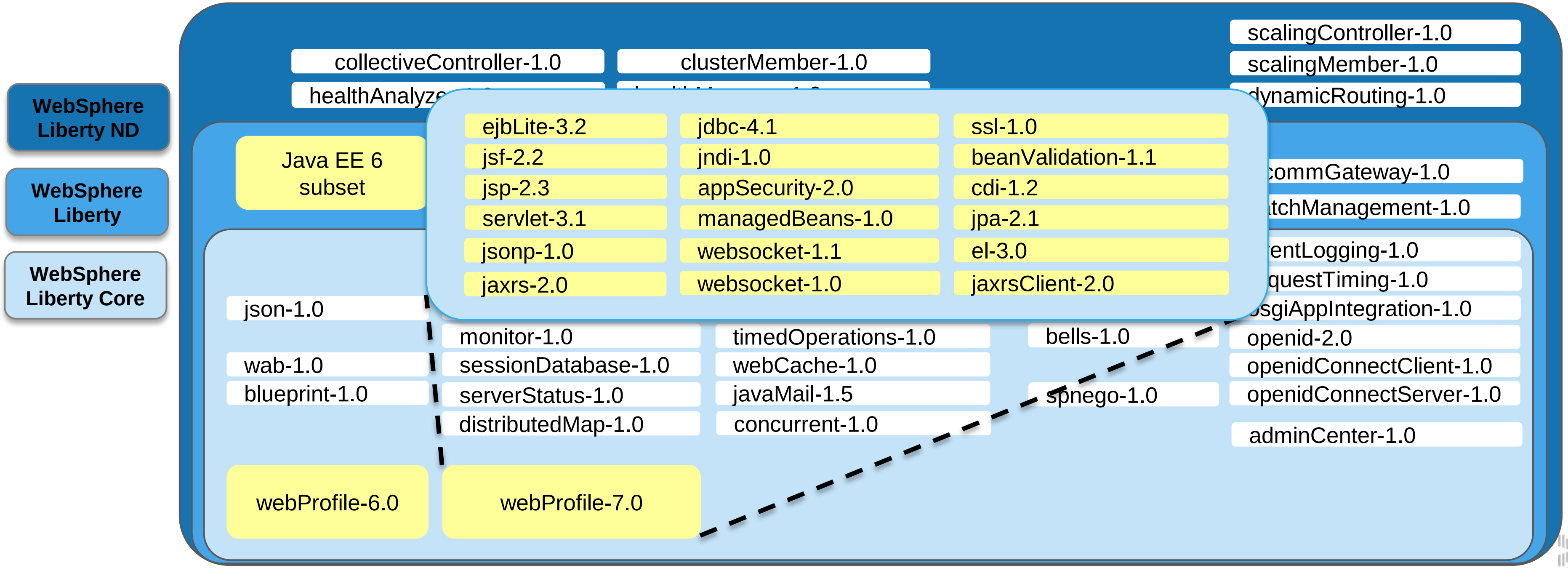
A Composable Java EE Runtime



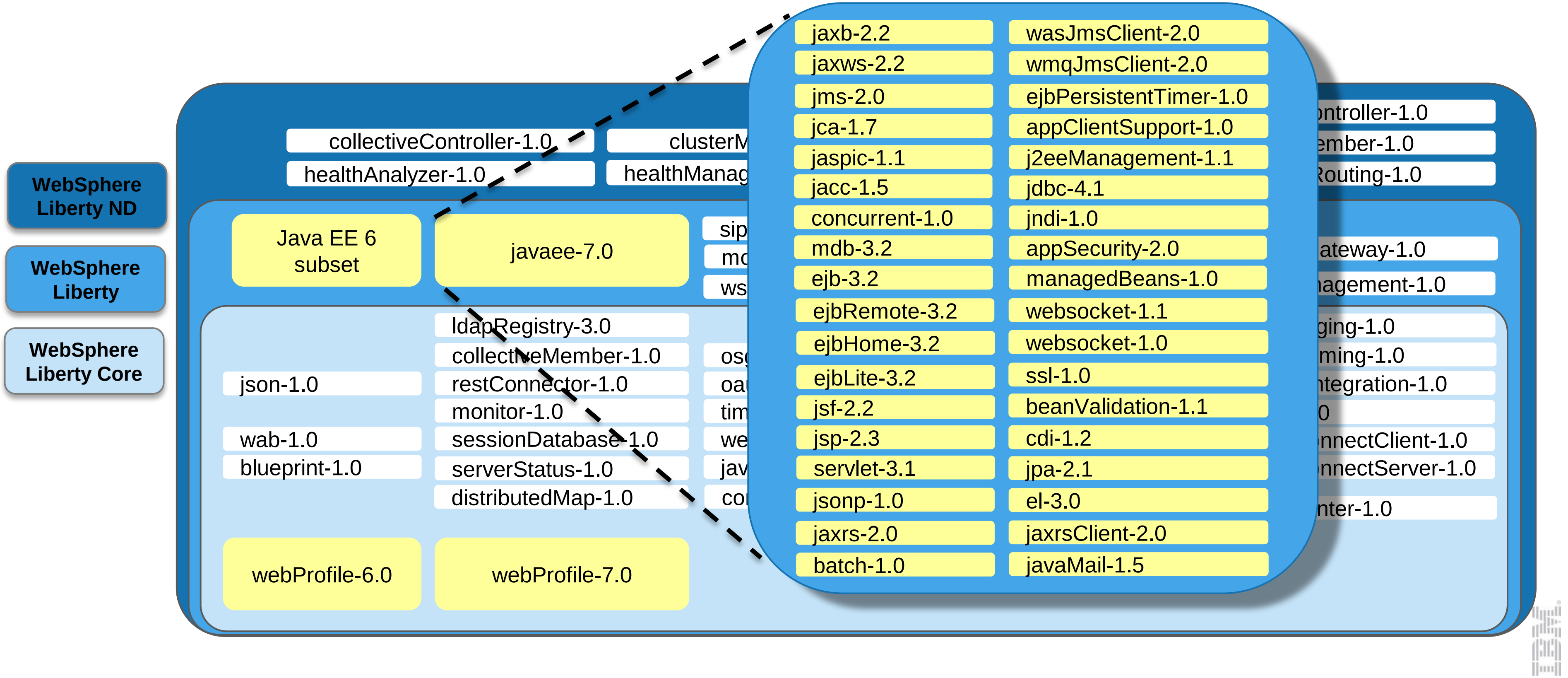
Composable Java EE 6 Web Profile



Composable Java EE 7 Web Profile



Fully Composable Java EE 7



Server Config: features, apps, resources

```
<server description="new server">

    <!-- Enable features -->
    <featureManager>
        <feature>servlet-3.1</feature>
        <feature>jdbc-4.1</feature>
    </featureManager>

    <webApplication id="blogapp“
          location="blogapp.war" name="blogapp"/>

    <include location="${shared.config.dir}/datasource.xml"/>
</server>
```

Features control which capabilities (bundles) are installed in the server

'instance' configurations specify multiple resources like applications and datasource definitions

Any of this configuration could be put into a separate xml file and 'included' in this 'master' configuration file

Barriers to Java EE*next* Adoption

#1 is Migration cost. Every spec breaking change and non-propagated bug costs our customers money

- An issue for EE vendors and framework-providers alike…

https://spring.io/blog/2015/06/10/feedback-welcome-spring-5-system-requirements
Spring 5 System Requirements Discussion, June 10, 2015

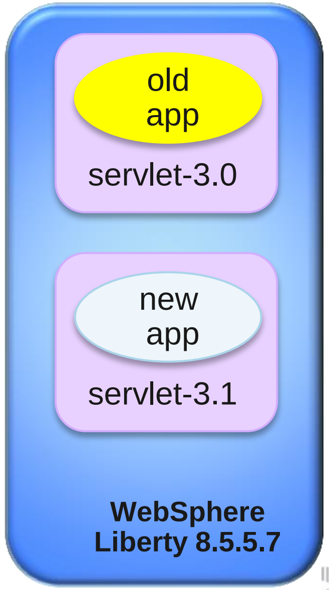
TL;DR

"We intend to softly upgrade the EE baseline as well.
Now, this is a bit tricky since we effectively have individual requirements here - and we need to consider the enterprise adoption levels in production environments:

- We'll definitely **raise to Servlet 3.0+** (from our present Servlet 2.5 runtime compatibility) but no higher since we'd like Spring 5 applications to run on EE 6 baselined servers still. See my previous blog post for a discussion on why this is unavoidable, given the market situation with Java EE 7 and the multitude of servers which is still based on the Servlet 3.0 API.
- **We'll keep our JMS 1.1+** compatibility since, aside from the EE 7 issue, we expect message brokers in the corporate world which are not necessarily upgraded to JMS 2.0 yet. Spring's JMS support automatically adapts to JMS 2.0 anyway, so there shouldn't be any lack in functionality. It's just a shame that we have to keep supporting the 2002-era JMS 1.1 API…
- We'd like to raise to JPA 2.1+ and Bean Validation 1.1+ **but our hands seem to be tied**: TomEE 1.7 and JBoss EAP 6.4 have hard **JPA 2.0** and **Bean Validation 1.0** APIs in them, and WebLogic 12.1.3 has JPA 2.1 but no Bean Validation 1.1 API (despite them being related).
- This means we'll have to keep detecting JPA 2.1 / BV 1.1, automatically adapting to them - **or we'll require local bundling of the JPA 2.1 / BV 1.1 API jars** and corresponding providers. A likely outcome is that we'll streamline our setup towards JPA 2.1, just tolerating JPA 2.0 at runtime through fallback checks, similar to how we handle Servlet 3.0 vs 2.5 at present."

The obvious solution is not that widely available.

- Although WebSphere has it ☺

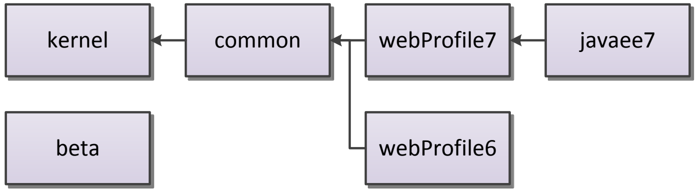


Is A Virtualization Container Really All We Need?

- A good unit of isolation
  - Full-stack delivery good for blue/green deploy.
  - High degree of control over container content
- But the App/Service in the virtualization container doesn't run in a vacuum.
  - Still an App Container of some description inside.
- Flowers and weeds take root **inside** the virtualization containers now…
- Java EE can be successful in this world but only if right-sized within the virtualization container.

Customized Docker containers for Java EE

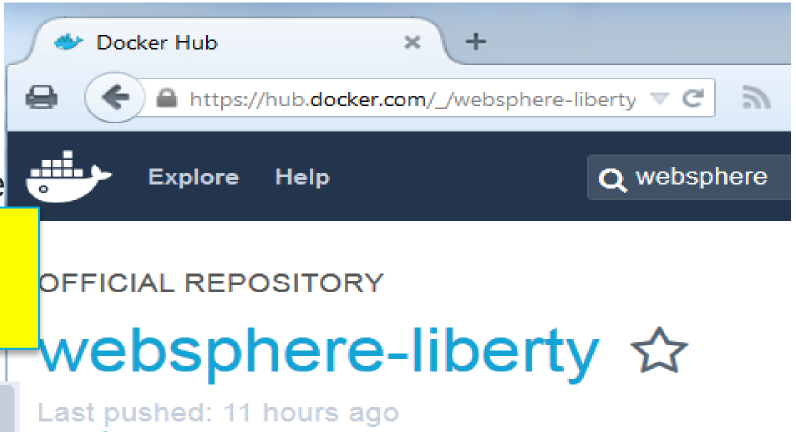
- WAS Liberty images on Docker Hub
  - WAS Liberty containers (currently V8.5.5.7):
    - Kernel, Java EE 6 Web Profile, Java EE 7 Web and Full Profile and latest Beta images
    - Docker files: https://github.com/WASdev/ci.docker



- Dockerfiles in on WASdev GitHub to:
  - Simple layer to upgrade to commercial license
  - Build your own customized image based on required feature

```
FROM websphere-liberty:kernel
COPY server.xml /opt/ibm/wlp/usr/servers/defaultServer/
RUN installUtility install defaultServer
```

hub.docker.com/_/websphere-liberty


Docker Hub

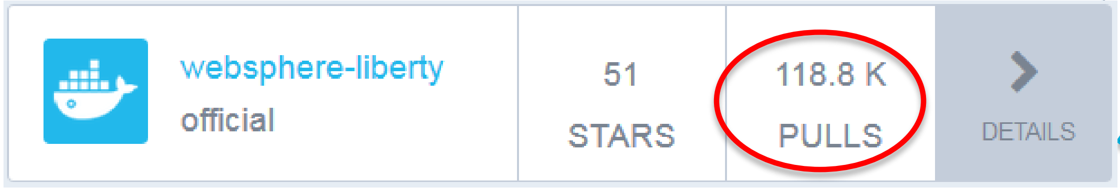
https://hub.docker.com/_/websphere-liberty

Explore Help websphere

OFFICIAL REPOSITORY

websphere-liberty

Last pushed: 11 hours ago


websphere-liberty official | 51 STARS | 118.8 K PULLS | DETAILS

Customized Docker containers for Java EE

```
FROM websphere-liberty:kernel
COPY server.xml /opt/ibm/wlp/usr/servers/defaultServer/
RUN installUtility install defaultServer
```

dockerfile

1. Want image for this

2. Identify required features

3. Add only the required features

www.wasdev.net

Java EE7 Web profile and Full platform: Additional components:

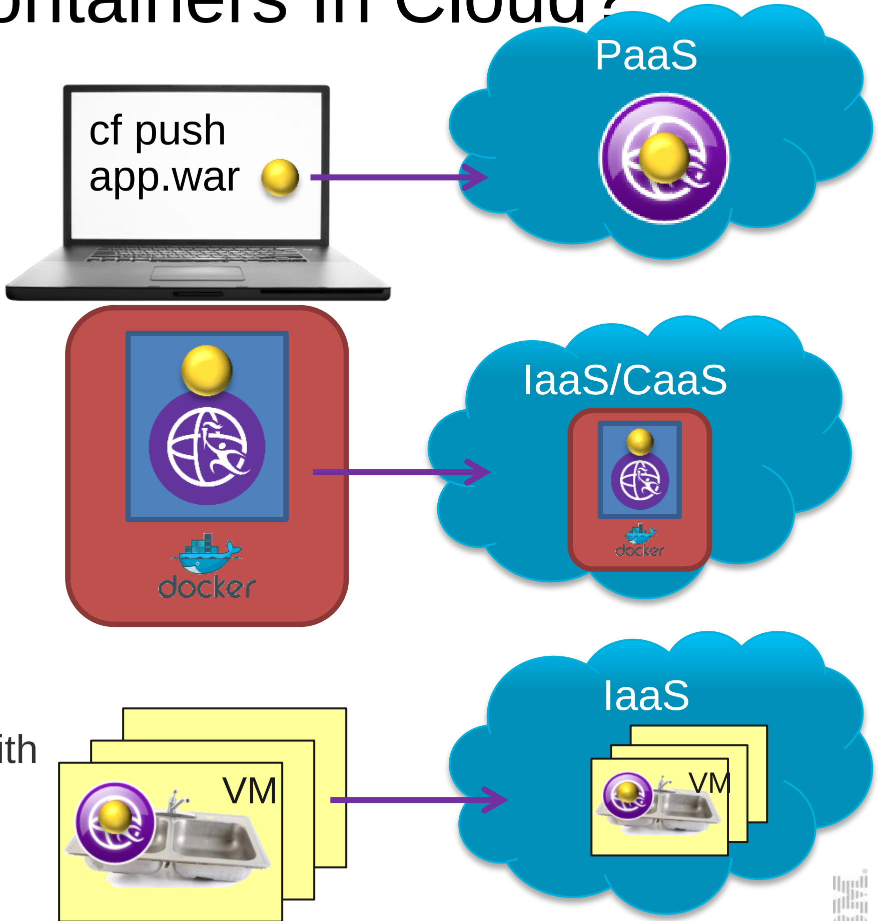
- ✓ Individual features
- ✓ Convenience features for EE6 EE7 web and full platform
- ✓ Zip archive product images
- ✓ Add-ons to bring an existing install up to Java EE7 web or full platform
- ✓ Beta drivers
- ✓ Deployable Samples with source in Github repo
- ✓ Config snippets

```
<server description="BlogServer">

    <!-- Enable features -->
    <featureManager>
        <feature>servlet-3.1</feature>
        <feature>jdbc-4.1</feature>
    </featureManager>

    ...
</server>
```

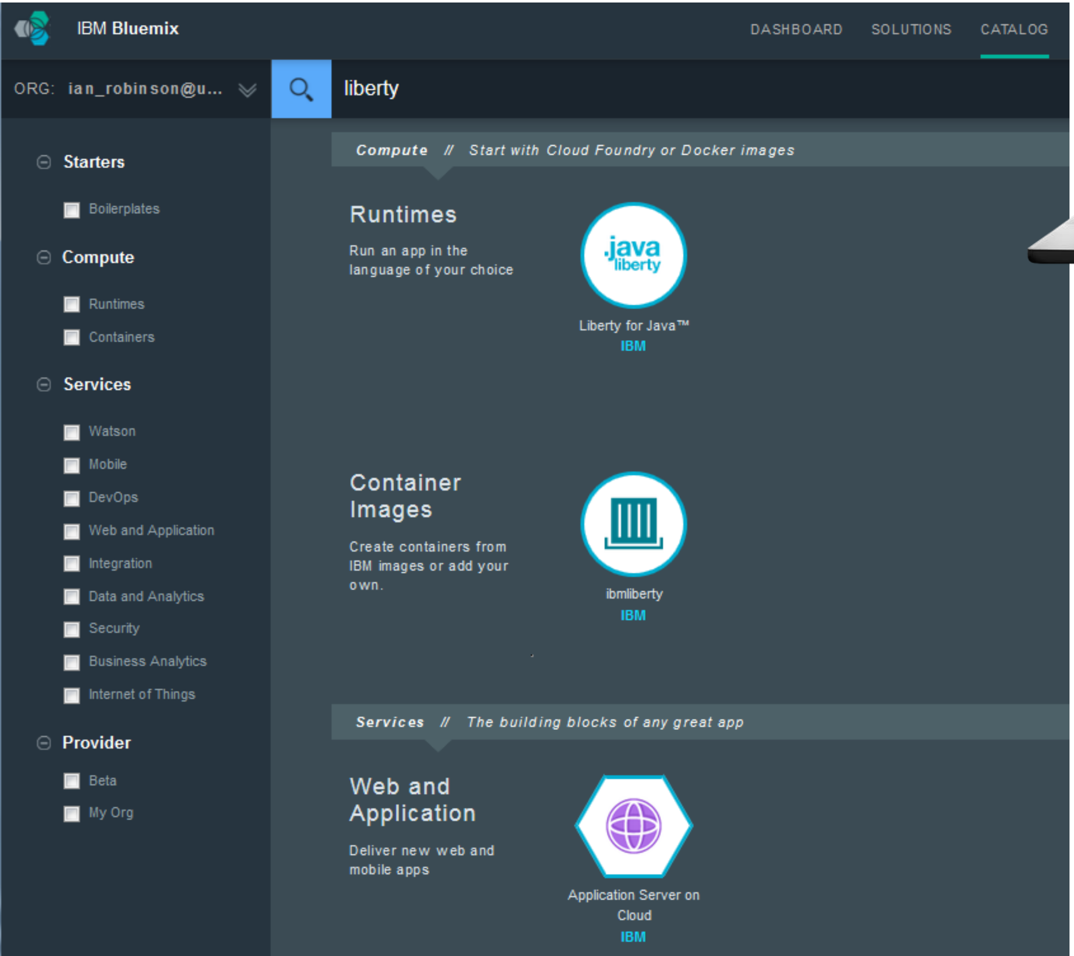
/opt/ibm/wlp/usr/servers/defaultServer/server.xml

Why Do I Care About App Containers In Cloud?

- Just push the Java App: the rest is a cloud detail
  - Could be any app container in the cloud
  - Configuration largely out of my hands
- If I need to customize the container config I can
  - But then I start to care about the app container
- For greater control, push the app container too
  - More stuff for me to own but greater control.
  - Better portability across clouds
- For greatest control, virtualize only at the IaaS
  - Easiest way to shift existing workloads to cloud with no change
- Only in the *very simplest case* do I not care about the app container.



IBM Bluemix Cloud and Java EE



IBM Bluemix

DASHBOARD SOLUTIONS CATALOG

ORG: ian_robinson@u...

liberty

- Starters
  - Boilerplates
- Compute
  - Runtimes
  - Containers
- Services
  - Watson
  - Mobile
  - DevOps
  - Web and Application
  - Integration
  - Data and Analytics
  - Security
  - Business Analytics
  - Internet of Things
- Provider
  - Beta
  - My Org

Compute // Start with Cloud Foundry or Docker images

Runtimes

Run an app in the language of your choice

Liberty for Java™
IBM

Container Images

Create containers from IBM images or add your own.

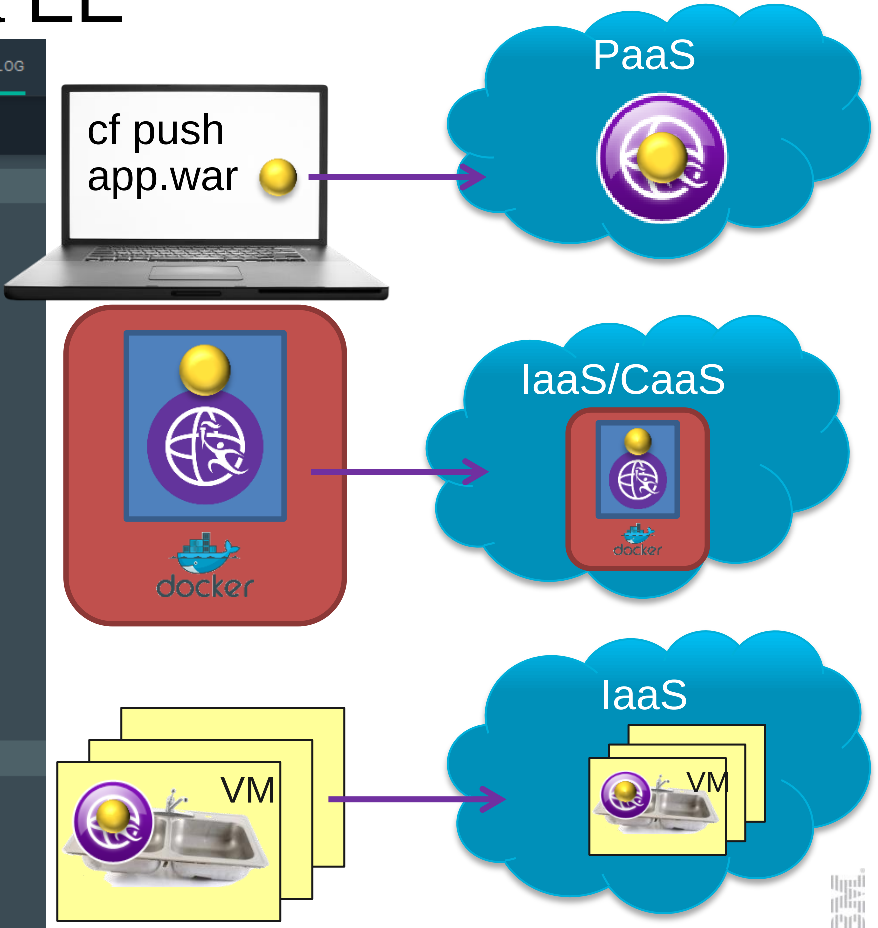
ibmliberty
IBM

Services // The building blocks of any great app

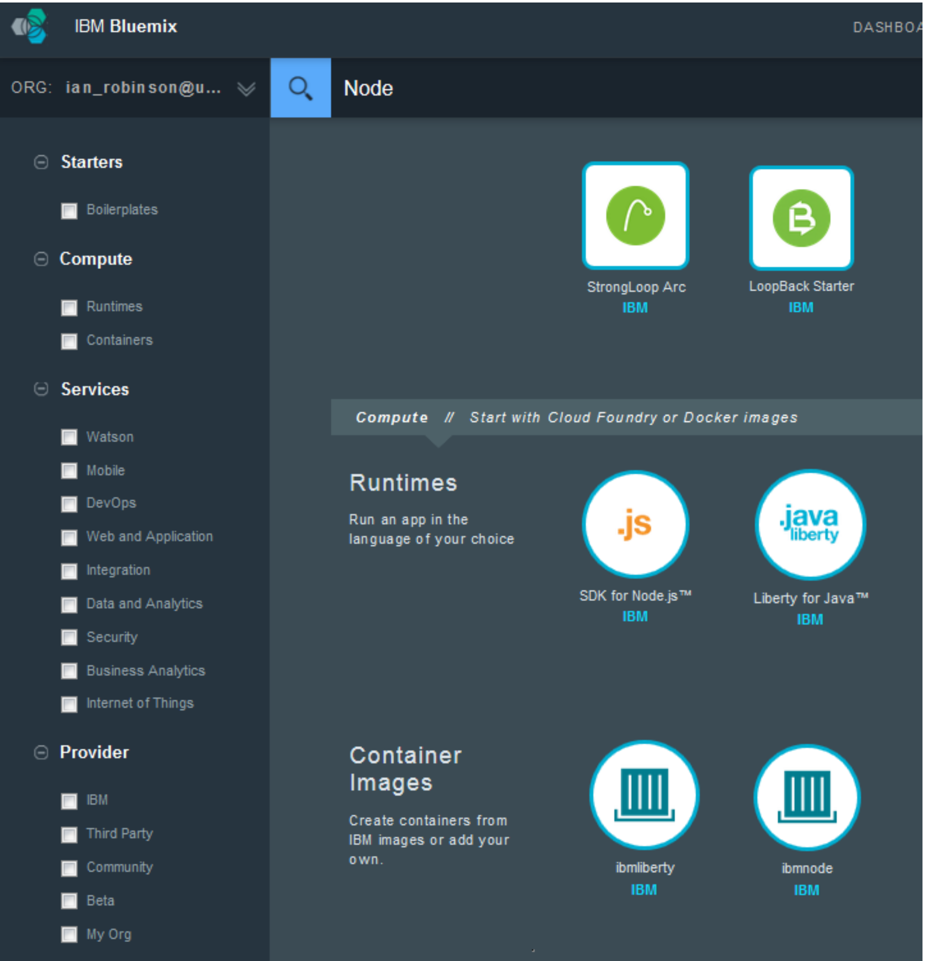
Web and Application

Deliver new web and mobile apps

Application Server on Cloud
IBM

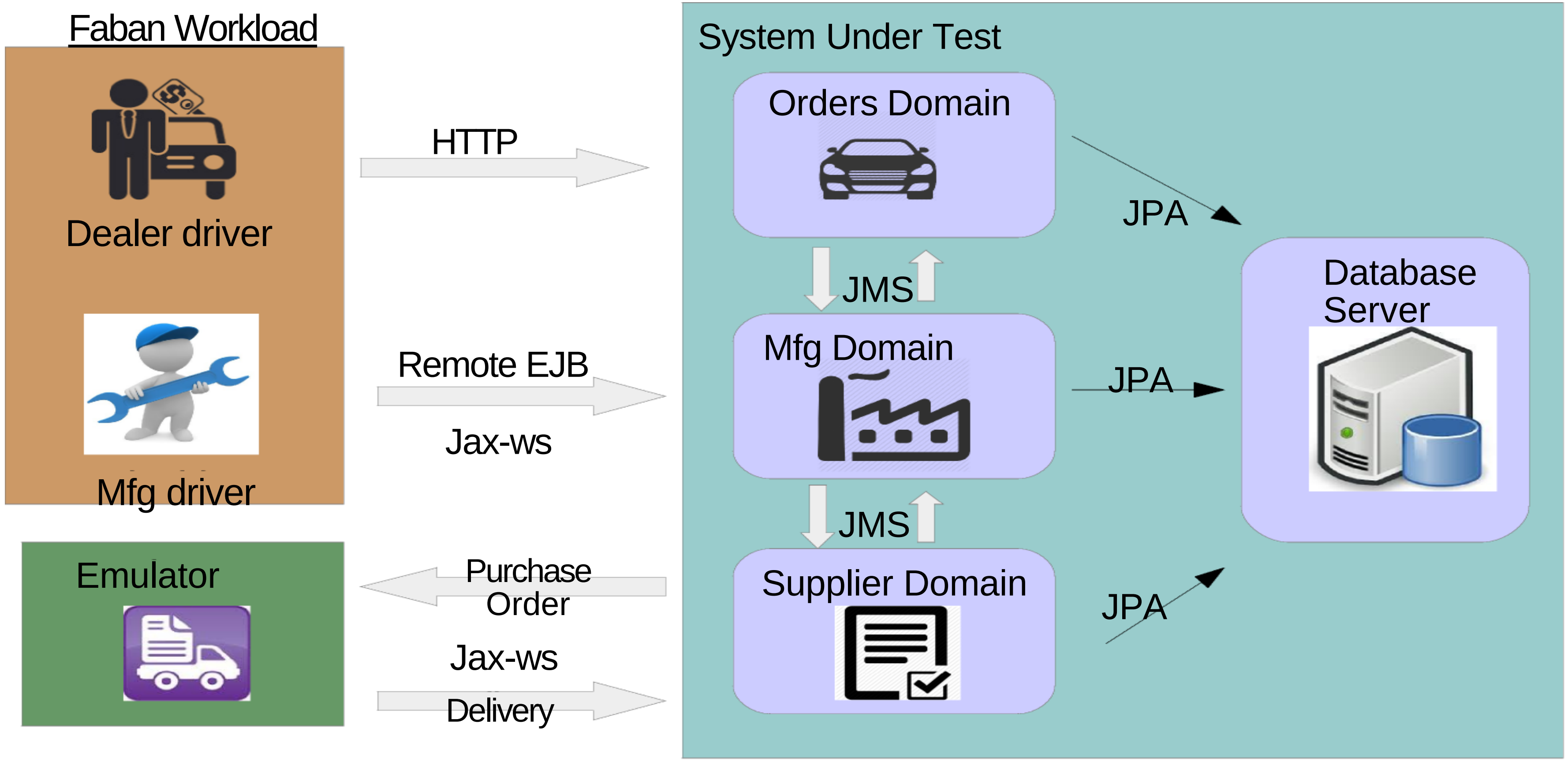


Java EE in a Multi-Cultural Society



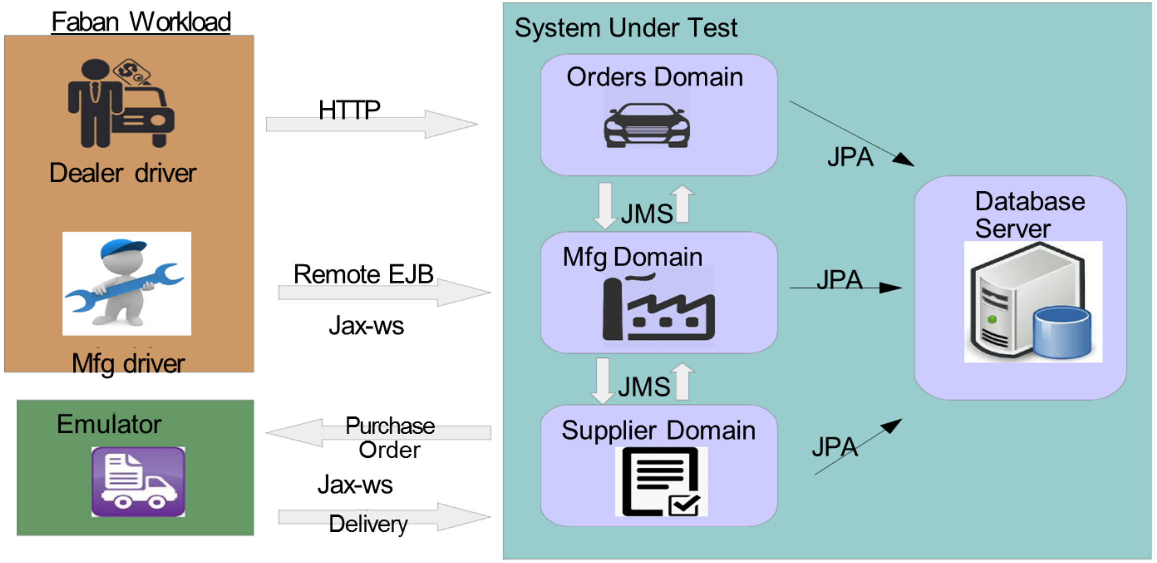
- Cloud unifies the management of different types of application container.
  - The 2 most popular runtimes in IBM Bluemix are Java and Node.js
    - Significantly ahead of others
- Node and other containers often compliment Java rather than replace it
  - E.g. using Node LoopBack framework to expose new APIs to existing Java services
- Cloud is providing new ways to deploy Java EE and driving additional workloads to it through multi-lingual API explosion
- Some commentators characterize cloud apps, mixing EE technologies with cloud services, as **Java-but-not-EE** applications.
  - Because they confuse Java EE with monolithic implementation. **CHALLENGE THIS!!**

Is This a Modern Application?



A Benchmark Living in the Past – SPECjEnterprise2010

- This is Java EE5 technology!
  - Not even EE6 let alone EE7.
- JCP Members are the primary contributors to this benchmark and are holding it back.
- **What are you scared of? More vendors?**
- Where is the web profile compliant subset?
- Where is EE7?
  - We can skip EE6…
- Run rules to allow cloud publish?



About Erin

- IBM Senior Software Engineer
- WebSphere Liberty architect
- Founding technical leader/developer for WAS Liberty
- Saturated in app-server internals and wire protocols (16+ years worth)
- Champion of composable runtimes, and object- or service-oriented approaches to decomposing complex systems
- Lives in Poughkeepsie, NY
- Family, kids, code, caffeine; all is well.

Erin Schnabel

Game On!

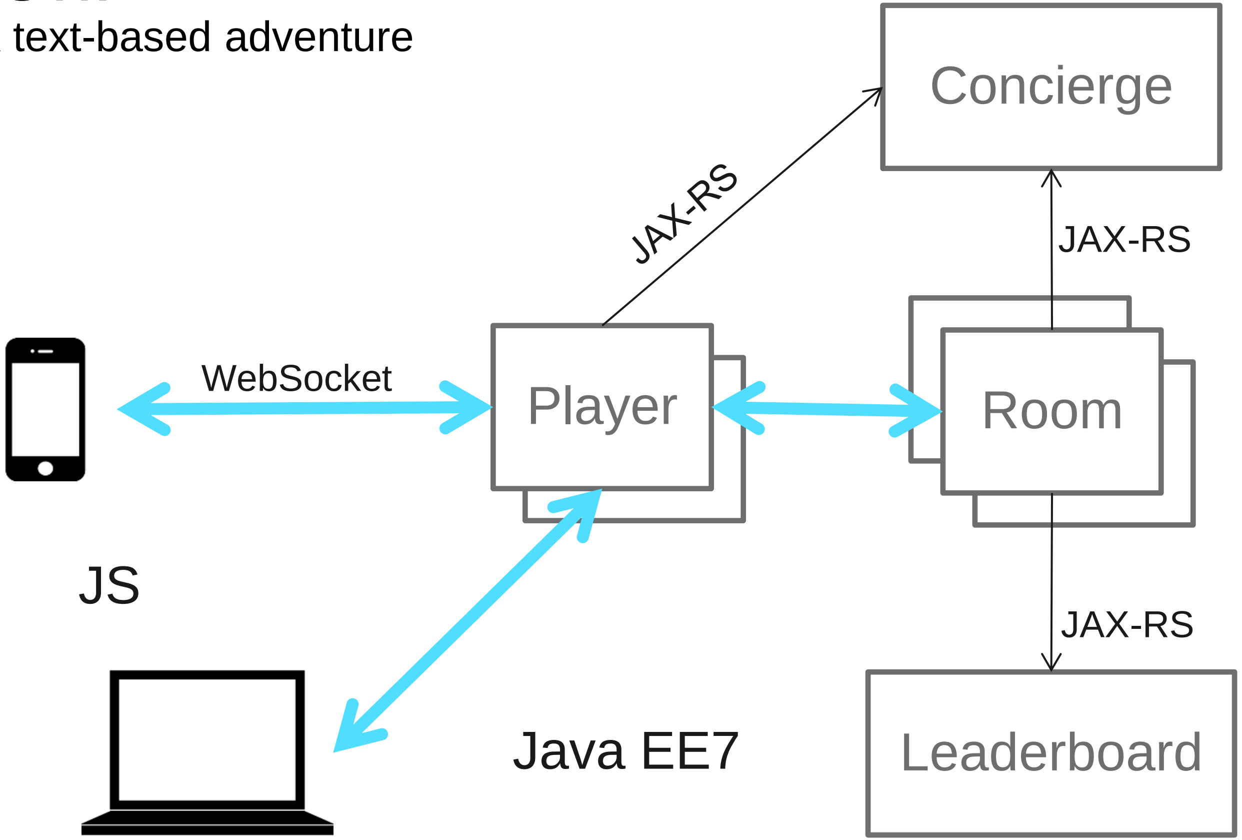
A throwback text-based adventure

You are in a maze of little interconnected rooms, none alike.

And you aren’t alone

Game On!

A throwback text-based adventure



Java EE Outlook

- Java EE is at a crossroads.
- Still strong and relevant at the cloud party, BUT younger guests are making more noise.
  - Vendors support an increasingly diverse cloud environment
- Enterprise incumbency = strength and weakness:
  - Customers want NEW without breaking EXISTING
  - NEVER break backward compatibility - sacrifices incumbency.
- Forward-looking EE 8 focus on web standards is good.
- Multi-tenancy? These days I'll use a virtualization container for that.
- And finally: stuff around the edges is critical
  - Still need to evangelize: SHOUT louder about lightweight Java EE
  - And lets get serious about a modern, lightweight EE7 benchmark

https://www.flickr.com/photos/akaitori

Thank You!

See us at the IBM Booth

Ian Robinson, IBM Distinguished Engineer, WebSphere Foundation Chief Architect
@ian__robinson

Erin Schnabel, IBM WebSphere developer/engineer/guru/evangelist
@ebullientworks