

InterConnect 2016

The Premier Cloud & Mobile Conference

BlueTag Mobile

A Journey from Prototype to a Complete Mobile
Networking App

PEJ-3670

Tom Seelbach

Muneeb Arshad

Vijai Kalathur



February 21 – 25
MGM Grand & Mandalay Bay
Las Vegas, Nevada

Please Note:

- IBM's statements regarding its plans, directions, and intent are subject to change or withdrawal without notice at IBM's sole discretion.
- Information regarding potential future products is intended to outline our general product direction and it should not be relied on in making a purchasing decision.
- The information mentioned regarding potential future products is not a commitment, promise, or legal obligation to deliver any material, code or functionality. Information about potential future products may not be incorporated into any contract.
- The development, release, and timing of any future features or functionality described for our products remains at our sole discretion.
- Performance is based on measurements and projections using standard IBM benchmarks in a controlled environment. The actual throughput or performance that any user will experience will vary depending upon many factors, including considerations such as the amount of multiprogramming in the user's job stream, the I/O configuration, the storage configuration, and the workload processed. Therefore, no assurance can be given that an individual user will achieve results similar to those stated here.

Abstract

Latest version will be here: <https://github.com/IBM-Bluemix/bluetag/tree/master/bluetag-docs>

BlueTag is a mobile application for finding people based on location, skills and interests in a conference setting. Think of it as a primitive Harry Potter "marauder's map" for geeks at a conference.

We will explain how we went from a prototype developed in a hackathon to iteratively developing the version we will demonstrate live.

The front-end leverages geo-location in iOS and Android and uses Apache Cordova.

The backend, built on Liberty and running in IBM Bluemix, is a set of microservices for location, registration and search. We will describe our technology choices and design for the frontend and the microservices. We will also describe our DevOps approach to build on Eclipse and deploy on Bluemix. Code is available on GitHub.

What you will learn:

Developers and architects will learn about key technologies used to develop a mobile location aware application using Apache Cordova in the frontend and micro services in the backend.

We will describe each of the micro services and explain why we choose JAX-RS for some, and Web Sockets in others. You will learn why we choose Cloudant as the database.

You will learn what it takes to run and monitor an application in BlueMix.

We will talk about "ramp-up" time to learn the new technologies.

Live demo!

Pressure is on YOU to make this work :)

<http://bluetag.mybluemix.net>



- Enter a name (or set of characters you identify with)
- Be sure to hit “allow” if prompted to enable location services
- You should appear on the map, along with others that are “near you”
- Tag someone in the near you list to add to your contacts.
- Search for any name, add them to your contacts.
- From Contacts, click the arrow to locate a contact on the map
(will only map people who are online, so ask a friend to join)

Bluetag Agenda

✓ What is it? Live demo

- Architectural overview

- About the journey

- Frontend explained

- Backend services explained

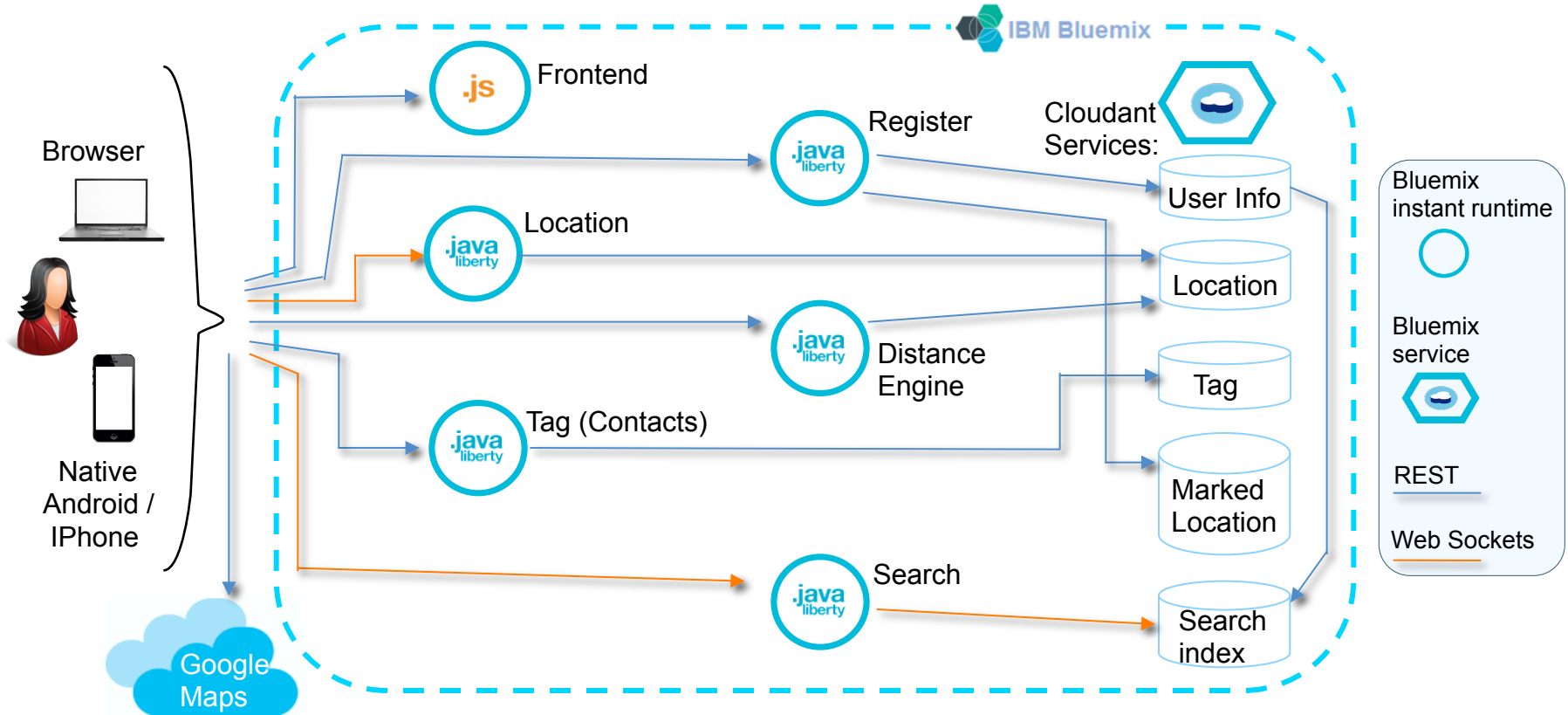
- Microservices defined

- 12 factor methodology

What it is, why it matters, and how well does Bluetag measure up

- Next steps for Bluetag

BlueTag Services overview



Bluetag APIs

Each of the services are available via API which is listed in the README. For example:

<https://github.com/IBM-Bluemix/bluetag/blob/master/bluetag-register/README.md>

Registration API: Check if the user passed in already exists in the database. If not, create an entry for that user.

Method: POST

URL: <bluetag-search service url>/api/register

Example URL: bluetag-register.mybluemix.net/api/register

```
Payload:  {
            "_id": "username"
            "name": "Actual Name"
        }
```

We intend to turn on Liberty apiDiscovery-1.0 feature so that you can see the live API's online

<https://developer.ibm.com/wasdev/blog/2016/02/17/exposing-liberty-rest-apis-swagger/>

Geolocation

We rely on the W3C recommended API – relatively consistent support in all browsers and all devices

<https://developer.mozilla.org/en-US/docs/Web/API/Geolocation>

App.js: *#farfromperfect #contributionWelcome*

```
navigator.geolocation.watchPosition(getWatchPositionSucess, getWatchPositionError, options);

function getWatchPositionSucess(pos) {
  var crd = pos.coords;
  console.log('getWatchPosition callback: current position is lat,lon,alt,accuracy(meters):
    ' + crd.latitude + ',' + crd.longitude + ',' + crd.altitude + ',' + crd.accuracy);

  ... update the map
  document.querySelector('#g-map').latitude = latitude;
  document.querySelector('#g-map').longitude = longitude;

  ... Save the location via microservice WebSocket call, store in Cloudant
  locationSocket.send(userlocJSON);

var options={enableHighAccuracy: true, timeout: 60000, maximumAge: 10000};
```


InterConnect 2016
The Premier Cloud & Mobile Conference

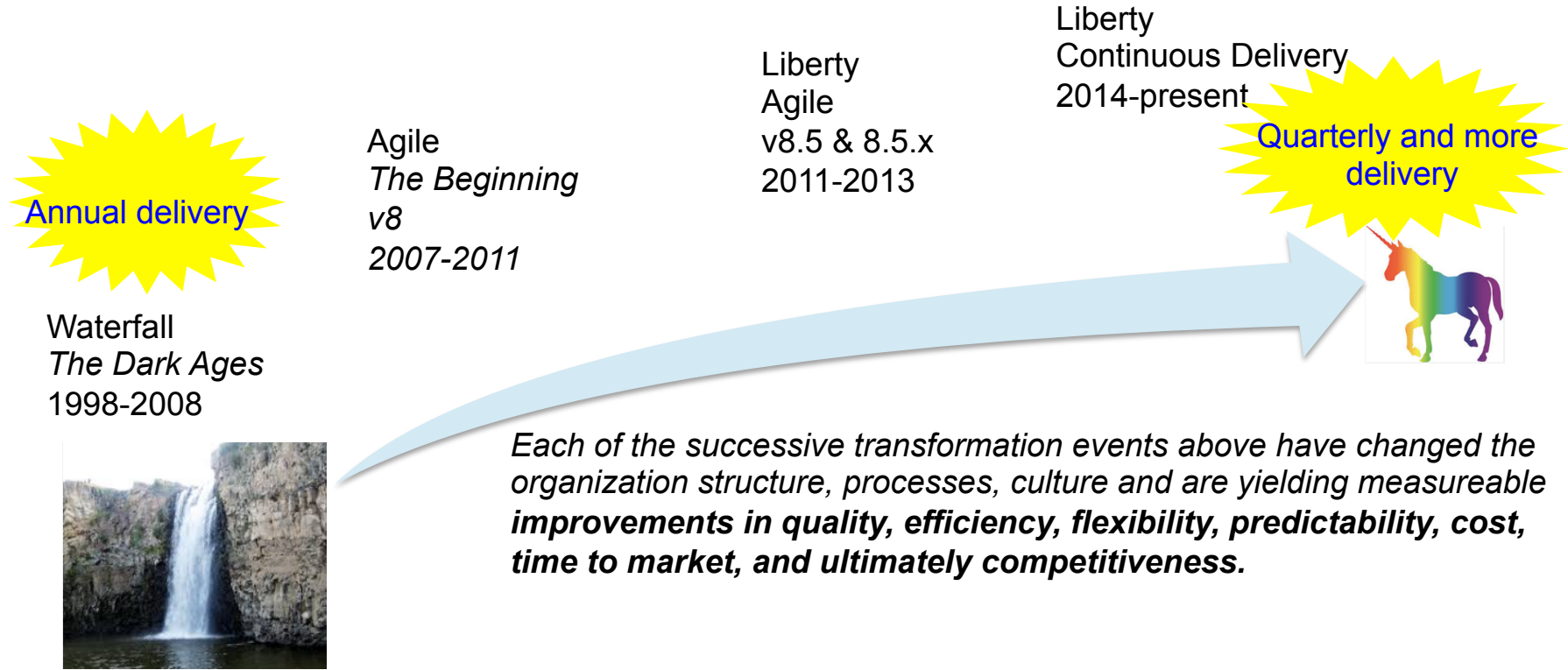
Bluetag development journey

Adventures in dis(joint)-agile



February 21 – 25
MGM Grand & Mandalay Bay
Las Vegas, Nevada

WebSphere's *On-going* Seven+ Year Agile Journey



Bluetag Journey – disjoint agile

- 1) leap of faith by our leadership: Go forth and be cloudy!
What's a hackathon? Mix a few students, distracted WAS developers, little background in subject area, shake well...
Zombie tag(seed for Bluetag) got second place? Only second?
 - 2) dis-agile development: (distracted) You want to be agile but you have x day jobs.
 - 3) (anecdotal) You best not be an IBMer with several old IBM Id's in the closet.
That will confuse the login gods and totally hose your BlueMix access
 - 4) nothing new here : frontend, middle tier, backend, good to go. But wait...
Ha what? I've got to gulp this and (cf) push that. Is that a node? Better get that checked...
A few more npm installs should do it... Oh fix it with this new Polymer we've discovered. Don't forget Bower update... What's the boundary between node server and js frontend?
 - 5) Go Native or go home – Anyone got an iPhone?
...Android: tap your heels 7 times on the build number and say “there's no place like a debug port via USB”

...Can I vulcanize that for ya? Nah I'll just browserify it thanks anyway
...So I just dump any old doc in the database, no structure required. Even with 1Million entries I can get one back fast – trust me ☺
... how many SDKs is enough?
- 1) it starts with leap of faith – thanks boss (no sucking up here ☺)

InterConnect 2016
The Premier Cloud & Mobile Conference

Bluetag Frontend

Adventures in Javascript land 😊



February 21 – 25
MGM Grand & Mandalay Bay
Las Vegas, Nevada

Frontend explained

Front end is a set of Web Components, built on Polymer, using the Starter Kit

<https://developers.google.com/web/tools/polymer-starter-kit/?hl=en>

<http://webcomponents.org/>

Closely follows W3C standards for

- Custom Elements
- HTML Imports
- Templates
- Shadow Dom

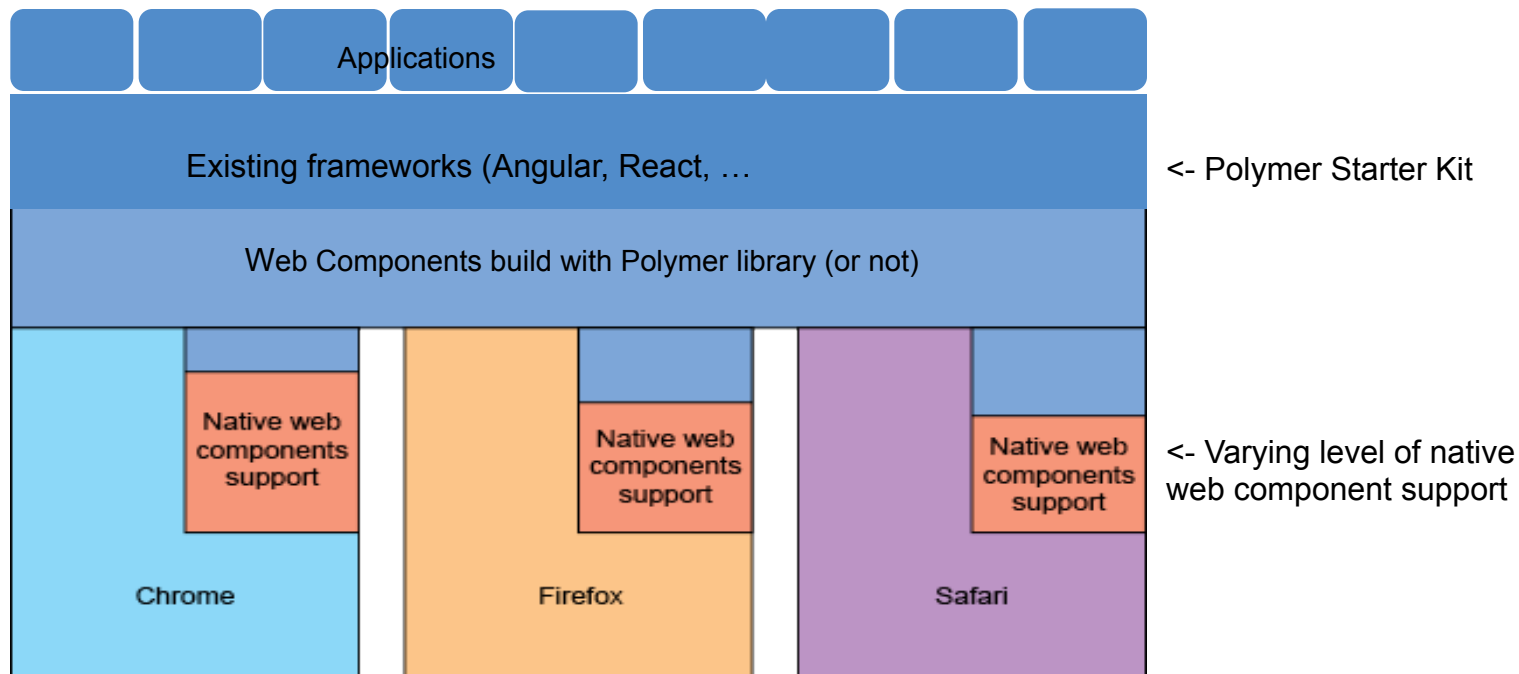
Polymer is a sugaring layer

- provides consistent Web Component implementation across browsers and devices

Provides a catalog of standard elements, implementing Google material design and more

<https://elements.polymer-project.org/>

Polymer overview



Web Component Standards

Web Components specifications

The key enabling draft specifications for the W3C Web Components standard are:

- [Custom Elements](#)
- [Shadow DOM](#)
- [HTML Imports](#)
- [HTML Templates](#)

Check out the [HTML components](#) series on developerWorks for more information.

See <http://www.ibm.com/developerworks/library/wa-polymer/>

- Good overview of Webcomponents, pointers to the specs
- Programming example is outdated because its based on Polymer 0.5
Polymer 1.0 is a major step forward.

Polymer, Angular, React, Oh My!!!

Polymer is a Web Component library, not a framework

- Could be used with React and Angular

- Helps smooth the transition while browser vendors evolve/implement Web Component Standards

Angular - an opinionated framework for building web apps

React - even more opinionated and structured

- Frameworks provide fast on ramp, but make integration with other apps tricky

- Web Component are a basic building block so in theory can be mixed/matched and reused more easily

Anyone want to write a BlueTag frontend in Angular? Would be a good comparison

Reference:

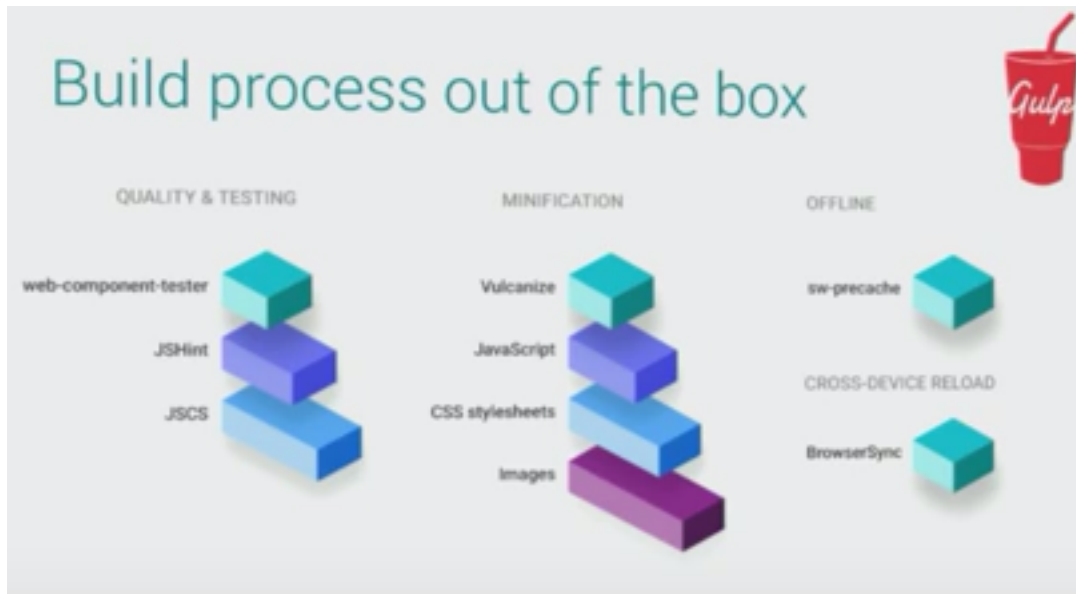
- Good Polymer and Web Component overview + using Polymer elements in Angular 2:

- Componentize your app with Polymer: <https://www.youtube.com/watch?v=7WgEuNZCCHk>

Polymer starter kit build process

Automated build process from the start – based on **Gulp**

- Critical for rapid UI development



From Rob Dodson at Angular U

<https://www.youtube.com/watch?v=7WgEuNZCCHk> minute 31:37

InterConnect 2016

The Premier Cloud & Mobile Conference

Bluetag backend

Adventures in micro-services 😊

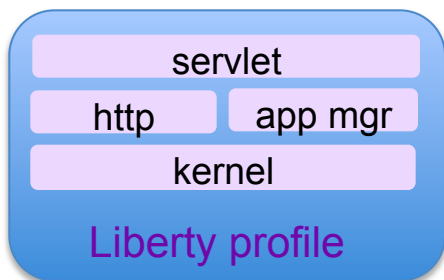


February 21 – 25
MGM Grand & Mandalay Bay
Las Vegas, Nevada

Backend services explained

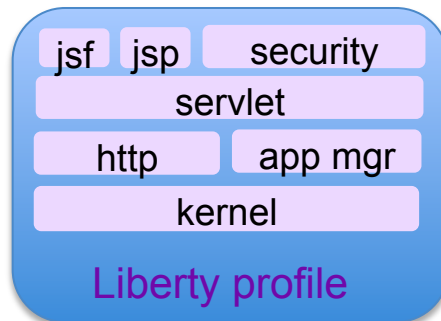
- Based on Liberty composeable server instances
- You control which features are loaded into each server instance

`<feature>servlet-3.0</feature>`



`<feature>jsf-2.0</feature>`

`<feature>appSecurity-2.0</feature>`



Internal feature dependencies are managed for you, for example:

- jsf feature includes jsp
- jsp feature includes servlet
- jdbc feature includes transactions

Example: Bluetag search service configuration

Search service only needs jax-rs and websockets.

- Then mix in a few Bluemix features

Server.xml:

```
<server>
  <featureManager>
    <feature>jaxrs-1.1</feature>
    <feature>websocket-1.0</feature>
    <feature>icap:managementConnector-1.0</feature>
    <feature>appstate-1.0</feature>
  </featureManager>

  <application name='myapp' location='myapp.war' type='war' context-root='/' />
  <httpEndpoint host="*" httpPort="{port}" id="defaultHttpEndpoint"/>
  <webContainer extractHostHeaderPort="true" trustHostHeaderPort="true"/>
  <include location="runtime-vars.xml"/>
  <logging consoleLogLevel="INFO" logDirectory="{application.log.dir}"/>
  <!-- httpDispatcher enableWelcomePage="false" -->
  <applicationMonitor dropinsEnabled="false" updateTrigger="mbean"/>
  <config updateTrigger="mbean"/>
  <!-- appstate appName="myapp" markerPath="{home}/../liberty.state"-->
</server>
```

Your choices: Liberty features – including Java EE 7

zOS

ND

Base

Core

New in
3Q15

New in
2Q15

New in
1Q15

New in
4Q14

zosSecurity-1.0

zosTransaction-1.0

zosWlm-1.0

zosConnect-1.2

zosLocalAdapters-1.0

collectiveController-1.0

clusterMember-1.0

healthAnalyzer-1.0

healthManager-1.0

scalingController-1.0

scalingMember-1.0

dynamicRouting-1.0

Java EE 6
subset

javaee-7.0

sipServlet-1.0

mongodb-2.0

wsSecurity-1.1

rtcomm-1.0

rtcommGateway-1.0

couchdb-1.0

batchManagement-1.0

json-1.0

wab-1.0

blueprint-1.0

webProfile-6.0

ldapRegistry-3.0

collectiveMember-1.0

restConnector-1.0

monitor-1.0

sessionDatabase-1.0

serverStatus-1.0

distributedMap-1.0

webProfile-7.0

osgiConsole-1.0

oauth-2.0

timedOperations-1.0

webCache-1.0

javaMail-1.5

concurrent-1.0

samlWeb-2.0

bells-1.0

spnego-1.0

eventLogging-1.0

requestTiming-1.0

osgiAppIntegration-1.0

openid-2.0

openidConnectClient-1.0

openidConnectServer-1.0

adminCenter-1.0

Liberty features – Java EE 6 Web Profile

zOS

ND

Base

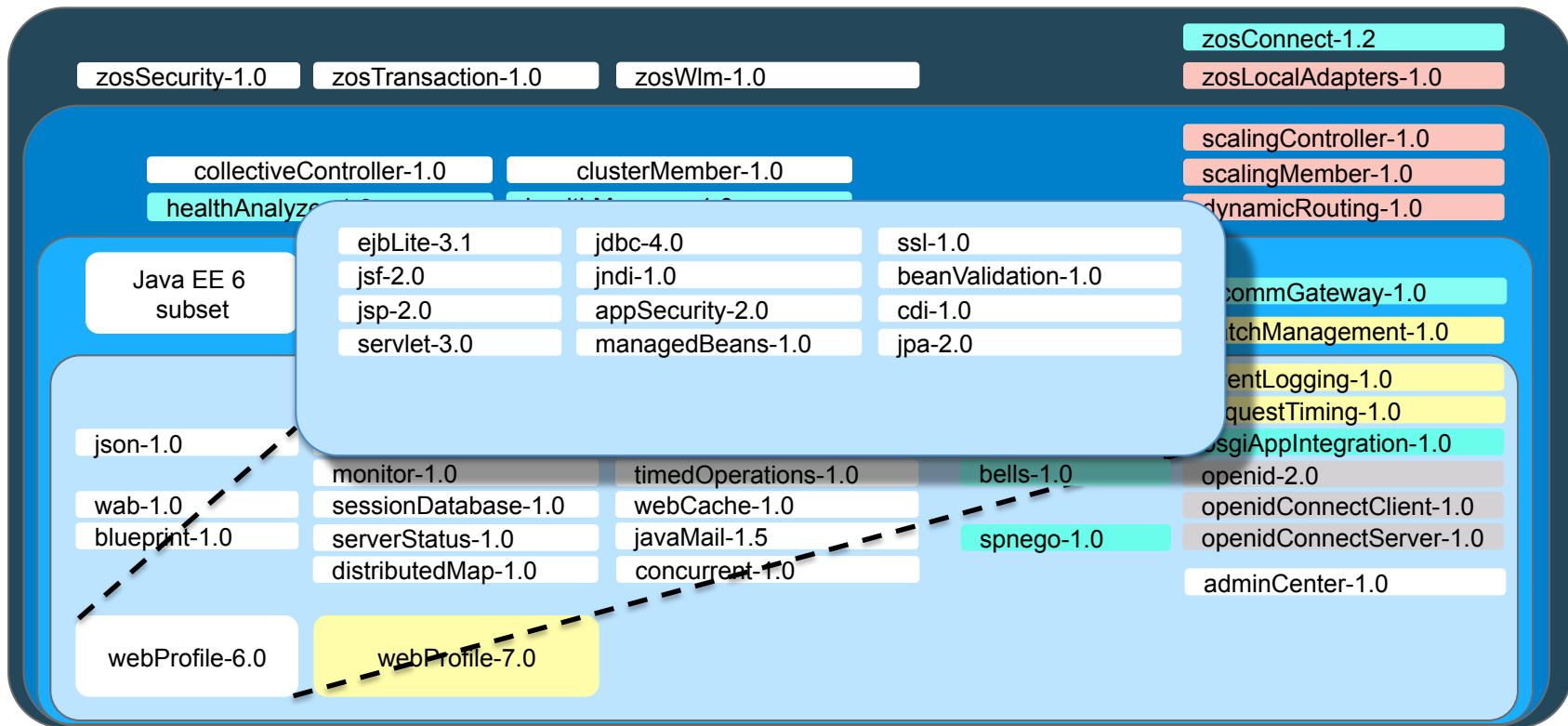
Core

New in
3Q15

New in
2Q15

New in
1Q15

New in
4Q14



Liberty features – Java EE 7 Web profile

zOS

ND

Base

Core

New in
3Q15

New in
2Q15

New in
1Q15

New in
4Q14

zosSecurity-1.0

zosTransaction-1.0

zosWlm-1.0

zosConnect-1.2

zosLocalAdapters-1.0

collectiveController-1.0

clusterMember-1.0

healthAnalyzer-1.0

scalingController-1.0

scalingMember-1.0

dynamicRouting-1.0

Java EE 6
subset

ejbLite-3.2

jdbc-4.1

ssl-1.0

jsf-2.2

jndi-1.0

beanValidation-1.1

jsp-2.3

appSecurity-2.0

cdi-1.2

servlet-3.1

managedBeans-1.0

jpa-2.1

jsonp-1.0

websocket-1.1

el-3.0

jaxrs-2.0

websocket-1.0

jaxrsClient-2.0

json-1.0

monitor-1.0

timedOperations-1.0

bells-1.0

wab-1.0

sessionDatabase-1.0

webCache-1.0

blueprint-1.0

serverStatus-1.0

javaMail-1.5

spnego-1.0

distributedMap-1.0

concurrent-1.0

webProfile-6.0

webProfile-7.0

commGateway-1.0

atchManagement-1.0

entLogging-1.0

requestTiming-1.0

jsgiApplIntegration-1.0

openid-2.0

openidConnectClient-1.0

openidConnectServer-1.0

adminCenter-1.0

Liberty features – Java EE 6 subset

zOS

ND

Base

Core

New in
3Q15

New in
2Q15

New in
1Q15

New in
4Q14

zosSecurity-1.0

zosTransaction-1.0

zos

collectiveController-1.0

clusterM

healthAnalyzer-1.0

healthManag

Java EE 6
subset

javaee-7.0

sip
mc
ws

json-1.0

wab-1.0

blueprint-1.0

webProfile-6.0

ldapRegistry-3.0

collectiveMember-1.0

restConnector-1.0

monitor-1.0

sessionDatabase-1.0

serverStatus-1.0

distributedMap-1.0

webProfile-7.0

jaxb-2.2

jaxws-2.2

jms-1.1

jca-1.6

mdb-3.1

jsf-2.0

jsp-2.2

servlet-3.0

jaxrs-1.1

wasJmsClient-1.1

wmqJmsClient-1.1

jdbc-4.0

jndi-1.0

appSecurity-2.0

managedBeans-1.0

ssl-1.0

beanValidation-1.0

cdi-1.0

jpa-2.0

connect-1.2

Adapters-1.0

controller-1.0

member-1.0

Routing-1.0

gateway-1.0

management-1.0

gging-1.0

ming-1.0

integration-1.0

0

connectClient-1.0

connectServer-1.0

inter-1.0

Liberty features – Java EE 7 Full Platform

zOS

ND

Base

Core

New in
3Q15

New in
2Q15

New in
1Q15

New in
4Q14

zosSecurity-1.0

zosTransaction-1.0

zos

collectiveController-1.0

clusterM

healthAnalyzer-1.0

healthManag

Java EE 6
subset

javaee-7.0

sip

mc

ws

json-1.0

ldapRegistry-3.0

collectiveMember-1.0

restConnector-1.0

monitor-1.0

sessionDatabase-1.0

serverStatus-1.0

distributedMap-1.0

os

oa

tim

we

jav

co

webProfile-6.0

webProfile-7.0

jaxb-2.2

wasJmsClient-2.0

connect-1.2

jaxws-2.2

wmqJmsClient-2.0

Adapters-1.0

jms-2.0

ejbPersistentTimer-1.0

controller-1.0

jca-1.7

appClientSupport-1.0

member-1.0

jaspic-1.1

j2eeManagement-1.1

Routing-1.0

jacc-1.5

jdbc-4.1

concurrent-1.0

jndi-1.0

mdb-3.2

appSecurity-2.0

gateway-1.0

ejb-3.2

managedBeans-1.0

management-1.0

ejbRemote-3.2

websocket-1.1

ging-1.0

ejbHome-3.2

websocket-1.0

ming-1.0

ejbLite-3.2

ssl-1.0

integration-1.0

jsf-2.2

beanValidation-1.1

0

jsp-2.3

cdi-1.2

connectClient-1.0

servlet-3.1

jpa-2.1

connectServer-1.0

jsonp-1.0

el-3.0

inter-1.0

jaxrs-2.0

jaxrsClient-2.0

batch-1.0

javaMail-1.5

Microservices defined

- Application architected as a suite of small services, each running in its own process, and communicating with lightweight mechanisms e.g. REST/HTTP
- Services built around business capabilities
- Each service independently deployable via automation
- Minimal centralised governance
 - May be written in different languages
 - May use different data storage technologies

Microservice compared to monolithic application

	Monolithic	Microservice
Architecture	Built as a single logical executable (typically the server-side part of a three tier client-server-database architecture)	Built as a suite of small services, each running separately and communicating with lightweight mechanisms
Modularity	Based on language features	Based on business capabilities
Agility	Changes to the system involve building and deploying a new version of the entire application	Changes can be applied to each service independently
Scaling	Entire application scaled horizontally behind a load-balancer	Each service scaled independently when needed
Implementation	Typically written in one language	Each service implemented in the language that best fits the need
Maintainability	Large code base intimidating to new developers	Smaller code base easier to manage
Transaction	ACID	BASE

12factors and Microservices

Hand and glove, peanut butter and jelly, ...



February 21 – 25
MGM Grand & Mandalay Bay
Las Vegas, Nevada

12-factor app methodology to gauge microservice-ee-ness

From <http://12factor.net> :

The twelve-factor app is a methodology for building software-as-a-service apps that:

- Use **declarative formats** for setup automation, to minimize time and cost for new developers joining the project;
- Have a **clean contract with the underlying operating system**, offering maximum portability between execution environments;
- **Designed for deployment on cloud platforms**, obviating the need for servers and systems administration;
- **Minimize divergence between development and production**, enabling continuous deployment for maximum agility;
- **Can scale up** without significant changes to tooling, architecture, or development practices.

Applies to apps written in **any programming language**, using **any combination of backing services** (database, queue, memory cache, etc).

Methodology written by Adam Wiggins - Heroku founder

Developed and battle tested in Precambrian era (2012) – has withstood test of time!

Check out: <https://github.com/WASdev/sample.microservices.12factorapp>

12 factors – why it matters

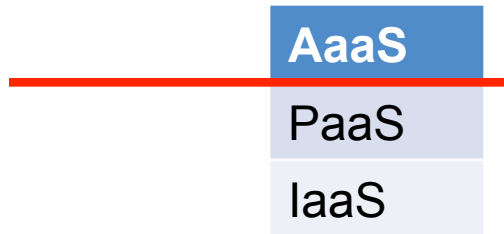
Clean, repeatable process is a critical success factor. Following the 12 factors is a way to gauge your probability of success.

Because architecting the code as services and using all the right tools may not produce good results.

<https://spring.io/blog/2015/01/30/why-12-factor-application-patterns-microservices-and-cloudfoundry-matter>

Tim Spann: *“Despite using Scrum, Cruise Control, SVN, Java, Eclipse, Guava, Google Guice, UML, JUnit, PMD, Findbugs, Checkstyle, MDD, TDD, eclEmma and mostly modern tools; our deploy process was a fragile, long, manual, person intensive process.”*

In the cloud, you could be running on a foo server,
...or bar – should not matter.



factor I – One codebase

- I. One codebase tracked in revision control, many deployments
- ✓ Bluetag is in Github, same code can be deployed locally, to staging, or to prod



Matters because: you cannot deploy with confidence if you don't have all the moving parts under control and coordinated

factor II - Explicitly declare and isolate dependencies

- ✓ Bluemix helps: manifest.yml declares dependencies on the service resources we need

```
applications:  
  - name: bluetag-register  
    memory: 512M  
    instances: 1  
    path: defaultServer  
    services:  
      - bluetag-cloudant  
    host: myOwn-bluetag-  
register
```

- ✓ Source code: we factored out some common libs, and models that are reused between the services
- ✓ Frontend: package.json declares gulp dependencies
env-config.json isolates the service dependencies from the code
- ✓ Backend: Isolated the Cloudant credential/service info



Why: Lets you deploy anywhere without code change

factor III - Store config in the environment

- ✓ Bluemix helps: `cf set-env` Lots of examples:
<https://www.ng.bluemix.net/docs/starters/liberty/index.html#customizingjre>



Frontend: `env-config.json` provides the values

But in `app.js` we pull in the file

```
var env = require('../env-config.json');
```



Backend: reads the `manifest.yml`

- ? May be worth creating a `github/bluetag/config` project, Store config files and script to push them to env




Config stored in env rather than files is less likely to get checked in and pushed to wrong deployment

factor IV - Treat backing services as attached resources

Bluetag backing services: Cloudant, Google Maps

- ✓ Attached via URL over the network
- ✓ Admin could spin up new Cloudant service, restore from backup, and change app attachment without code changes
- ✓ Clean API contract provided by Google Maps

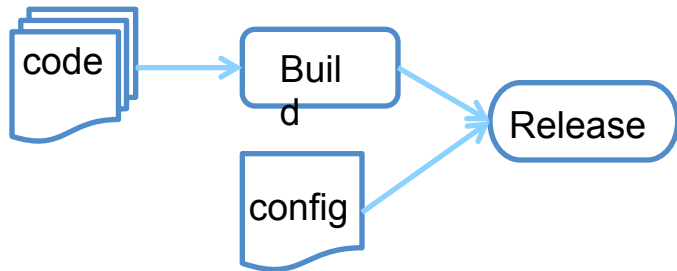
 No code change required to deploy to new environment
Backing services can be swapped out at will, quickly

factors V - Build, release, run: Strictly separate build and run stages

- ✓ Bluetag is comprised of independent micro-services with their own build/release/run process

Backend – build: create WAR
release: cf push
run: automatic cf restage

Frontend – build: gulp Serve
release: cf push
native build: cordova run
run: auto cf restage



- Build can be more complex. Run must be extremely simple
- Admin could spin up new Cloudant service, restore from backup, and change app attachment without code changes



- Provides clean separation of stages. Developers have control over build and (mostly) release. Run should be mostly hands off – minimizes middle of the night surprises
- Done right: Allows for rollbacks

factor VI – app executes as one or more stateless processes

- ✓ All Bluetag services are stateless and share nothing
- ✓ Persistence is concentrated in Cloudant backing service
- Todo: what about sessions? What is our cookie design?
 - Sticky sessions – caching user data in process – leads to dependencies / complexities in routing
 - We haven't rolled in authentication (user or API) and full lifecycle.
 - Need to make sure we don't introduce unnecessary state
 - Good candidates for caching services



Allows processes to come and go, scale out and in.
Removes dependencies between processes

factor VII - Export services via port binding

- ✓ Frontend: node.js server with configurable port. No code changes to run either:
<http://localhost:5000>
or
<http://mybluetag.myBluemix.net>
- ✓ Backend ports configured in server.xml:

```
...  
<application name='myapp' location='myapp.war' type='war'  
context-root='/' />  
    <httpEndpoint host="*" httpPort="${port}"  
id="defaultHttpEndpoint"/>  
...
```



Services are completely self contained. Allows service reuse and independence.

factor VIII – Concurrency, Scale apps out via the process model

Think of process conceptually as the Unix process model:

Can run a process (service) locally for dev, the run as a daemon for prod (with auto-respawn)

Model applies equally well to node.js process, java process, provided we keep the service contained using 12factors

Even though under the covers, the “process” may be multi-threaded, use controller/worker pattern, etc

Builds on the previous factors, especially:

Stateless

Bind services via ports

- ✓ Frontend: add more Node.js instance
 - ✓ Bluemix helps: `cf scale` But we haven't tried it out yet....
- ✓ Backend: All the java processes in theory can scale out similar to frontend.
- ✓ Cloudant: We have simple API interface to Cloudant. Scaling managed by Cloudant:
<https://cloudant.com/product/cloudant-features/scalability/>



By following 12factor methodology, we now reap the benefits when BlueTag goes viral 😊

factor IX – Disposability ...

... Maximize robustness with fast startup and graceful shutdown
Service processes can start/stop at moments notice
Facilitates rapid and frequent deployments

Again builds and depends on previous factors!

- ✓ Frontend/backend services: kill at will go ahead!
 - Momentary glitches may be visible
 - Don't know exactly what front end caching and service worker code will do
- ✓ Backend: Liberty launches in seconds – not ur grandpa's WebSphere
- ✓ Cloudant: hmm may want to test that
 - Is there a wrinkle in the fabric? How to test... hmmm chaos monkey sounds right
 - Have not thought about graceful shutdown. (i'm sure it will just work ☺)



Critical for agile: need to rev often with confidence to evolve!!!

Not just scaling, but elastic scaling based on immediate need

factors X - Keep dev, staging, and prod similar

Mind the gaps:

- Time from dev to prod : hours/days not weeks/months
 - People: Developers shepherd their code all the way to prod and monitoring
 - Tools: Use the same ones everywhere
- ✓ Bluemix helps: Spin up backing services (Cloudant) and instant runtimes with very little effort

Bluetag: N/A because we test in prod 😊 But if we were real...

- ✓ Time: in hours
- ✓ People: Dev responsible for prod
- ✓ Tools: All the same at least staging to prod. Dev can use a local webserver and/or Liberty
 - No automated tests

... We'd be awesome!



Prevents premature graying
Reduces cost (yawn)

Factor ... Are we really going thru all 12?

No way, can't be...

What session am I going to next... WAIT is this the last one ☺

Wow what an awesome concert !!!

... is he still talking? I'm hooked on this awesome stalker app...

Squirrel! Squirrel! Where!

<https://www.youtube.com/watch?v=xrAIGLkSMIs>



PAY ATTENTION! THIS IS IMPORTANT

factors XI - Treat logs as event streams

Apps should just write to stdout and not be concerned about log management.

- In Development, log events are visible in various consoles
- In Prod, events are aggregated by the runtime env and processed for many purposes:
monitor, debug, timing/performance, ...

- ✓ Bluemix/Liberty helps: logMet collector (Beta)
 - Collect and send events to logMet server
 - Google “liberty logmet feature” to get to WebSphere Knowledge Center article
`<feature>logmetCollector-1.0</feature>`

Bluetag:

- ✓ Yup – we just write to stdout.
- ✓ Simplest view in bluemix: `cf logs <service-name>` # tails the log
- Have not tried logMet yet



Many independent Microservices can be tricky to manage. Write the app simply, and rely on log aggregation and analytics

factors XII - Run admin/mgmt tasks as one-off processes

e.g. DB cleanup, migration, and query scripts

Think of admin tasks as similar to app service processes and follow the same 12 factor rules:

- checked in code, part of the release process
- developers can test the admin tools in their very similar dev and staging environments
- Bluetag: none so far
- ✓ Bluemix helps: the paas environment requires no maintenance out of the box. We just use it and it works



You don't want to blow up the world with a one-off script! Even with traditional applications, the more reliable and consistent the admin services, the less problems that come up

InterConnect 2016

The Premier Cloud & Mobile Conference

In conclusion...



February 21 – 25
MGM Grand & Mandalay Bay
Las Vegas, Nevada

Todo's and parting thoughts

Commercialize the app: Make #meelions ☺ Seriously: We tried to keep it real, an app we'd like to use...

- You should study more in depth microservices architecture and applications:

<https://game-on.org>

Architecture, patterns, and more:

<https://developer.ibm.com/architecture/gallery/createMicroservices>

TODO

- Devops toolchain for backend,
- Expose APIs via api-discovery-1.0
- Wrap in OAUTH authentication, make your ID,,, well,,, your ID (using game-on model)
 - pull in public profile info and expand search for skills/interests
- Switch to https
- Engage friend over SMS, perhaps using contacts on native, and/or Twilio APIs
- Stripped down version that just finds users (finduser - fu.mybluemix.net ☺)
- Exploration of social and privacy aspects. Pet name: "the stalker app"
 - re-enable the setting panel – add more user controls
- Add automated tests

Someday/maybe:

- learn about #ibeacon
- Put it on Play AppStore,
- flesh out native
- Scale out
- Ops dashboard – LogMet
- DB utils
- Make distance engine scalable
- push notifications
- ...

Please join us! Contribute, report issues, ask questions: <http://github.com/bluemix/bluetag/>

Notices and Disclaimers

Copyright © 2016 by International Business Machines Corporation (IBM). No part of this document may be reproduced or transmitted in any form without written permission from IBM.

U.S. Government Users Restricted Rights - Use, duplication or disclosure restricted by GSA ADP Schedule Contract with IBM.

Information in these presentations (including information relating to products that have not yet been announced by IBM) has been reviewed for accuracy as of the date of initial publication and could include unintentional technical or typographical errors. IBM shall have no responsibility to update this information. THIS DOCUMENT IS DISTRIBUTED "AS IS" WITHOUT ANY WARRANTY, EITHER EXPRESS OR IMPLIED. IN NO EVENT SHALL IBM BE LIABLE FOR ANY DAMAGE ARISING FROM THE USE OF THIS INFORMATION, INCLUDING BUT NOT LIMITED TO, LOSS OF DATA, BUSINESS INTERRUPTION, LOSS OF PROFIT OR LOSS OF OPPORTUNITY. IBM products and services are warranted according to the terms and conditions of the agreements under which they are provided.

Any statements regarding IBM's future direction, intent or product plans are subject to change or withdrawal without notice.

Performance data contained herein was generally obtained in a controlled, isolated environments. Customer examples are presented as illustrations of how those customers have used IBM products and the results they may have achieved. Actual performance, cost, savings or other results in other operating environments may vary.

References in this document to IBM products, programs, or services does not imply that IBM intends to make such products, programs or services available in all countries in which IBM operates or does business.

Workshops, sessions and associated materials may have been prepared by independent session speakers, and do not necessarily reflect the views of IBM. All materials and discussions are provided for informational purposes only, and are neither intended to, nor shall constitute legal or other guidance or advice to any individual participant or their specific situation.

It is the customer's responsibility to insure its own compliance with legal requirements and to obtain advice of competent legal counsel as to the identification and interpretation of any relevant laws and regulatory requirements that may affect the customer's business and any actions the customer may need to take to comply with such laws. IBM does not provide legal advice or represent or warrant that its services or products will ensure that the customer is in compliance with any law

Notices and Disclaimers Con' t.

Information concerning non-IBM products was obtained from the suppliers of those products, their published announcements or other publicly available sources. IBM has not tested those products in connection with this publication and cannot confirm the accuracy of performance, compatibility or any other claims related to non-IBM products. Questions on the capabilities of non-IBM products should be addressed to the suppliers of those products. IBM does not warrant the quality of any third-party products, or the ability of any such third-party products to interoperate with IBM's products. IBM EXPRESSLY DISCLAIMS ALL WARRANTIES, EXPRESSED OR IMPLIED, INCLUDING BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE.

The provision of the information contained herein is not intended to, and does not, grant any right or license under any IBM patents, copyrights, trademarks or other intellectual property right.

IBM, the IBM logo, ibm.com, Aspera®, Bluemix, Blueworks Live, CICS, Clearcase, Cognos®, DOORS®, Emptoris®, Enterprise Document Management System™, FASP®, FileNet®, Global Business Services®, Global Technology Services®, IBM ExperienceOne™, IBM SmartCloud®, IBM Social Business®, Information on Demand, ILOG, Maximo®, MQIntegrator®, MQSeries®, Netcool®, OMEGAMON, OpenPower, PureAnalytics™, PureApplication®, pureCluster™, PureCoverage®, PureData®, PureExperience®, PureFlex®, pureQuery®, pureScale®, PureSystems®, QRadar®, Rational®, Rhapsody®, Smarter Commerce®, SoDA, SPSS, Sterling Commerce®, StoredIQ, Tealeaf®, Tivoli®, Trusteer®, Unica®, urban{code}®, Watson, WebSphere®, Worklight®, X-Force® and System z® Z/OS, are trademarks of International Business Machines Corporation, registered in many jurisdictions worldwide. Other product and service names might be trademarks of IBM or other companies. A current list of IBM trademarks is available on the Web at "Copyright and trademark information" at: www.ibm.com/legal/copytrade.shtml.

Thank You

InterConnect 2016
The Premier Cloud & Mobile Conference

Your Feedback is Important!

Access the InterConnect 2016 Conference Attendee
Portal to complete your session surveys from your
smartphone,
laptop or conference kiosk.



February 21 – 25
MGM Grand & Mandalay Bay
Las Vegas, Nevada