

Jakarta EE 和 MicroProfile 的明天会怎样？

Emily Jiang

Liberty Lead Architect for CDI, MicroProfile

Java Champion

TGO 鲲鹏会

汇聚全球科技领导者的高端社群

📍 全球12大城市

👤 850+ 高端科技领导者

使命
Mission

为社会输送更多优秀的
科技领导者

愿景
Vision

构建全球领先的有技术背景
优秀人才的学习成长平台



扫描二维码，了解更多内容

About Emily

- Java Champion
- STSM, IBM, Liberty Lead Architect for MicroProfile and CDI
- Leads MicroProfile Config, Fault Tolerance, Service Mesh
- Co-spec lead for Config JSR
- CDI Expert Group
- Based in IBM's Hursley lab, UK

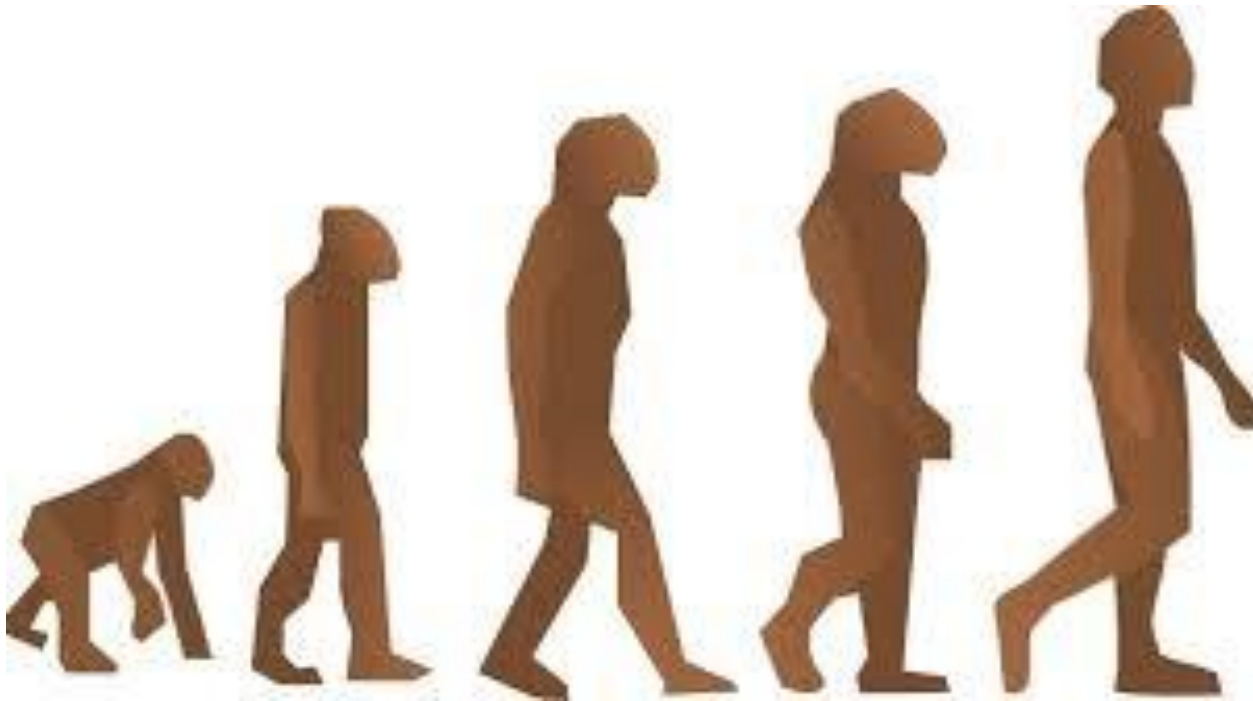
Open Liberty



Emily Jiang
IBM

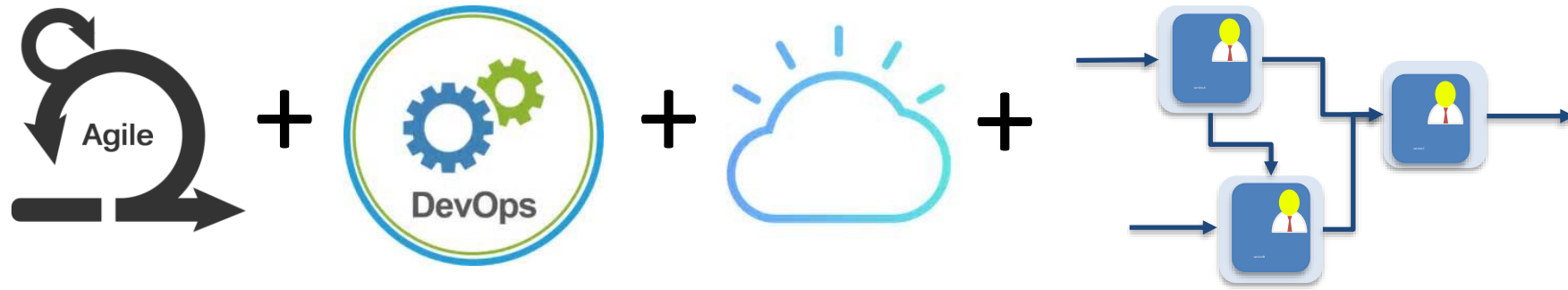
IT evolution

Open Liberty



Agile & DevOps & Cloud & Microservices

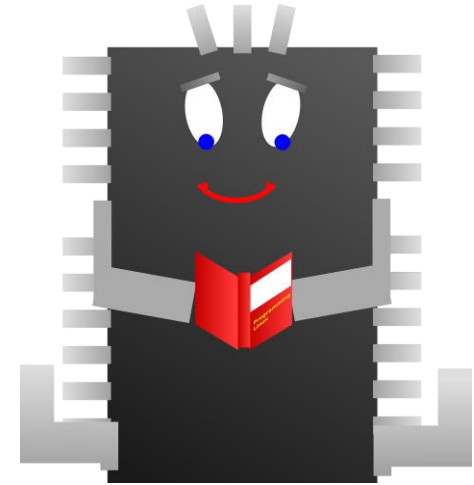
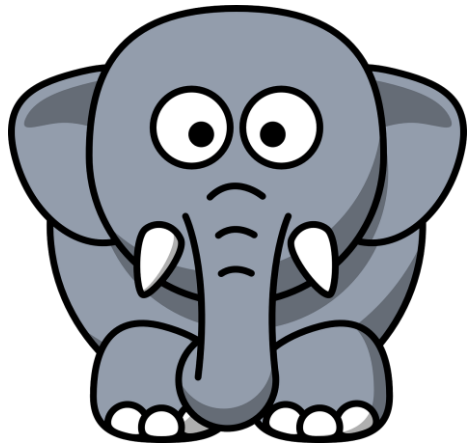
Open Liberty



Cloud is the future











- Small runtime memory footprint
- Small deployment sizes
- Fast starting applications
- No resource usage when idle





cloud-native microservice

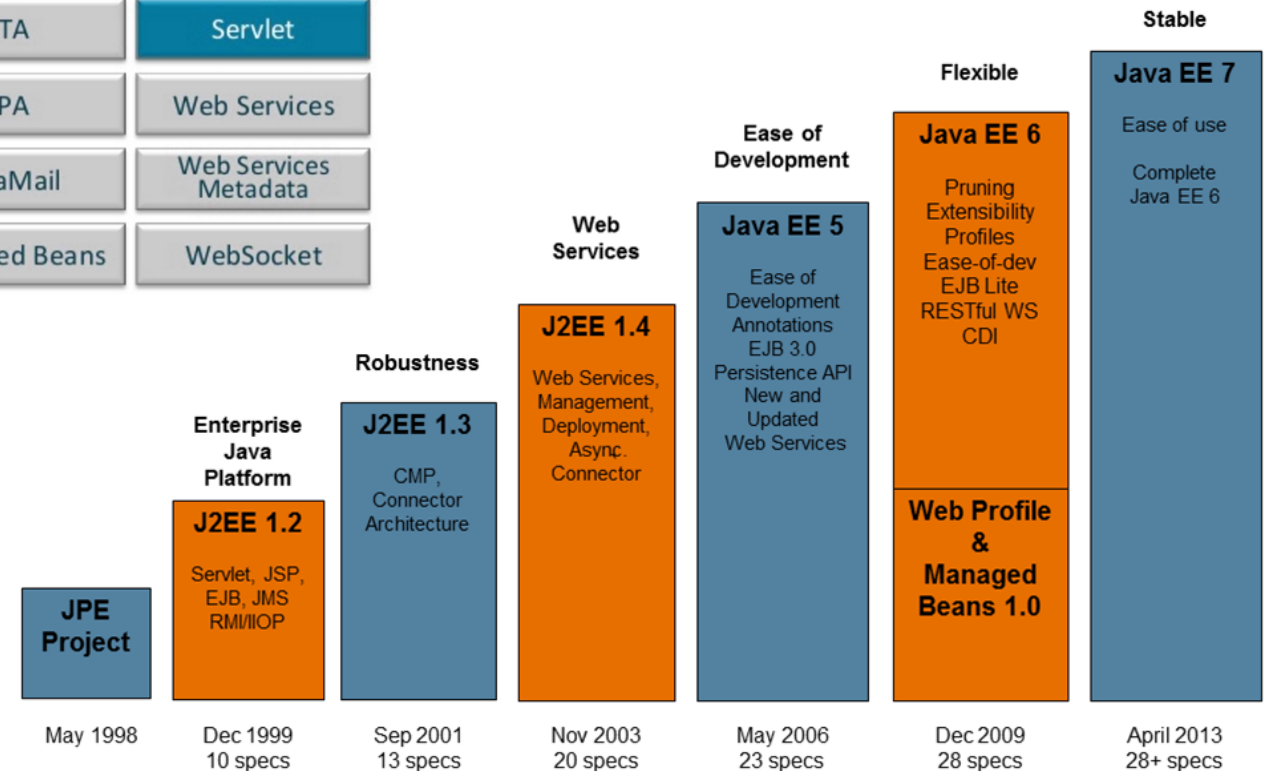
1. RESTful – like cattle not pet, communicative 
2. Configurable 
3. Fault tolerance 
4. Can be discovered 
5. Secure 
6. Traceable, monitorable  
7. Able to communicate with the cloud infrastructure 

Java EE's too slow and bloated, right?

Open Liberty



Batch	Dependency Injection	JACC	JAXR	JSTL	Management
Bean Validation	Deployment	JASPIC	JMS	JTA	Servlet
CDI	EJB	JAX-RPC	JSF	JPA	Web Services
Common Annotations	EL	JAX-RS	JSON-P	JavaMail	Web Services Metadata
Concurrency EE	Interceptors	JAX-WS	JSP	Managed Beans	WebSocket
Connector	JSP Debugging	JAXB			
JSON-B	Security				



<https://www.slideshare.net/delabassee/java-ee-8-february-2017-update>
<https://medium.com/@alextheedom/java-ee-past-present-future-8bf25df7b6a3>

How can we help?

Open Liberty



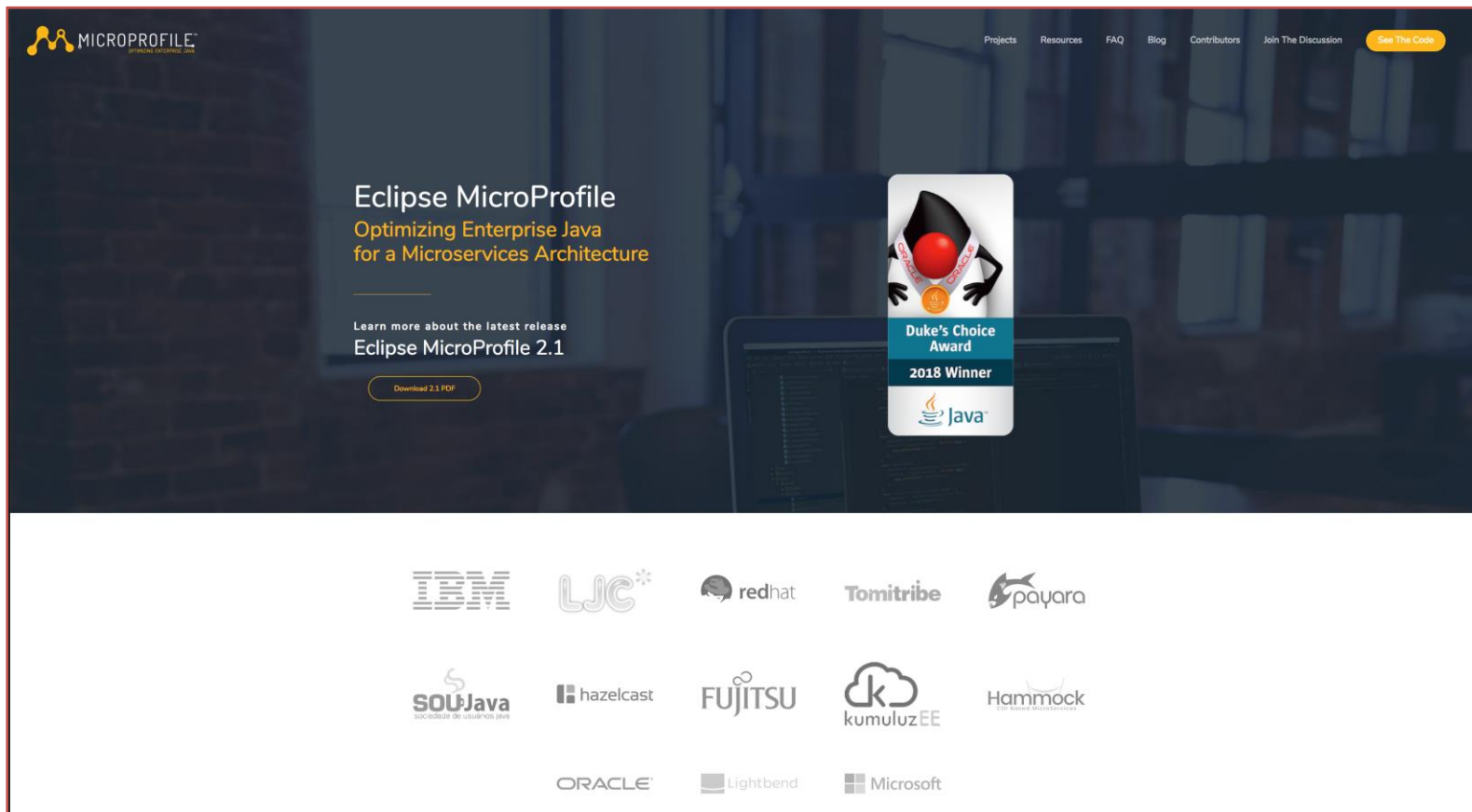
What can we do to advance microservice development in the Enterprise Java space?

-Java EE Community, early 2016

9

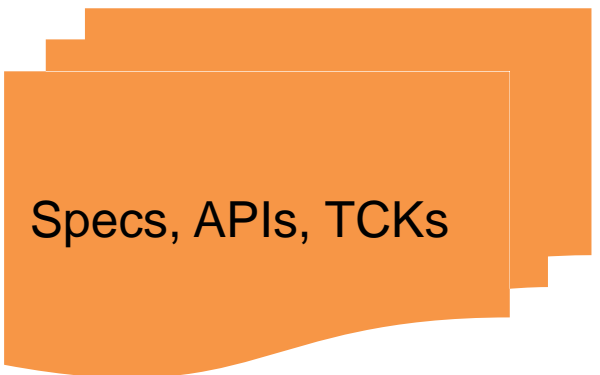


MICROPROFILETM
OPTIMIZING ENTERPRISE JAVA



Community
Driven

NO Reference Implementation



JavaOne 2016

Open Liberty



**MicroProfile 1.0
Announced!**

CDI 1.2

JAX-RS 2.0

JSON-P 1.0

Basic Building Blocks for Microservices

11

Fast-forward two years...

MicroProfile
2.2

Open Liberty



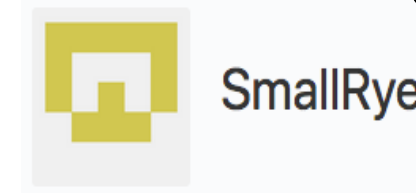
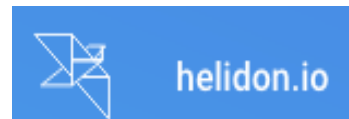
Open Tracing 1.3	Open API 1.1	Rest Client 1.2	Config 1.3
Fault Tolerance 2.0	Metrics 1.1	JWT Propagation 1.1	Health Check 1.0
CDI 2.0	JAX-RS 2.1	JSON-P 1.1	JSON-B 1.0

MicroProfile **2.2**

8 Platform Releases!

21
Component Releases!

Open specifications
Wide vendor support
REST services
OpenAPI support
Security
Fault Tolerance
Configuration
Metrics
Health
Open Tracing



Quarkus

<https://wiki.eclipse.org/MicroProfile/Implementation>



Open Liberty

19.0.0.2

19.0.0.1

19.0.0.3

Jan

MicroProfile 1.1 (August 2017)

microProfile-1.0
mpConfig-1.0

2017

MicroProfile 1.0 (Fall 2016)

jaxrs-2.0
cdi-1.2
jsonp-1.0

MicroProfile 1.2 (Sept 2017)

microProfile-1.1
mpConfig-1.1
mpFaultTolerance-1.0
mpHealth-1.0
mpMetrics-1.0
mpJwt-1.0

MicroProfile 1.4 (June 2018)

MicroProfile 1.3
mpConfig-1.3
mpFaultTolerance-1.1
mpJwt-1.1
mpOpenTracing-1.1
mpRestClient-1.1

2018

MicroProfile 1.3 (Dec 2017)

MicroProfile 1.2
mpConfig-1.2
mpMetrics-1.1
mpOpenApi-1.0
mpOpenTracing-1.0
mpRestClient-1.0

MicroProfile 2.2 (Feb 2019)

Fault Tolerance 2.0
OpenAPI 1.1
OpenTracing 1.3
Rest Client 1.2

MicroProfile 2.1 (Oct 2018)

MicroProfile 2.0
mpOpenTracing-1.2

MicroProfile 2.0.1 (July 2018)

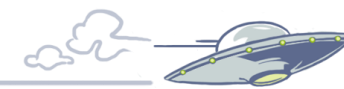
MicroProfile 1.4
jaxrs-2.1 // Java EE 8
cdi-2.0 // Java EE 8
jsonp-1.1 // Java EE 8
jsonb-1.0 // Java EE 8



There's a good chance you'll use **REST APIs**

Eclipse MicroProfile

Open Liberty



JAX-RS

Rest Client

CDI

JSON-P

JSON-B



Tomitribe



ORACLE®



microprofile.io

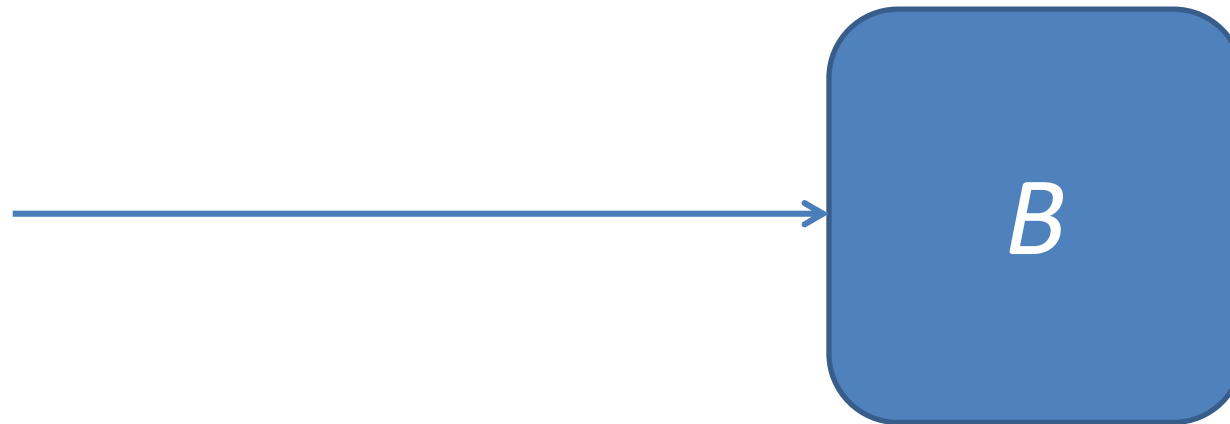


FUJITSU



JAX-RS

Open Liberty



```
@ApplicationPath("System")  
public class SystemApplication extends Application {}
```

```
@Path("properties")  
public class PropertiesResource {  
  
    @GET  
    @Produces(MediaType.APPLICATION_JSON)  
    public JsonObject getProperties() {...}  
  
}
```

MicroProfile REST Client

Open Liberty



```
@Dependent
@registerRestClient
@registerProvider(UnknownUrlExceptionHandler.class)
@Path("/properties")
public interface SystemClient {
    @GET
    @Produces(MediaType.APPLICATION_JSON)
    public Properties getProperties() throws UnknownUrlException,
        ProcessingException;
}
```

```
@Inject
@RestClient
private SystemClient defaultRestClient;
```

```
io.openliberty.guides.inventory.client.SystemClient/mp-rest/url=http://localhost:9080/system
```

CDI - Contexts Dependency Injection

Open Liberty

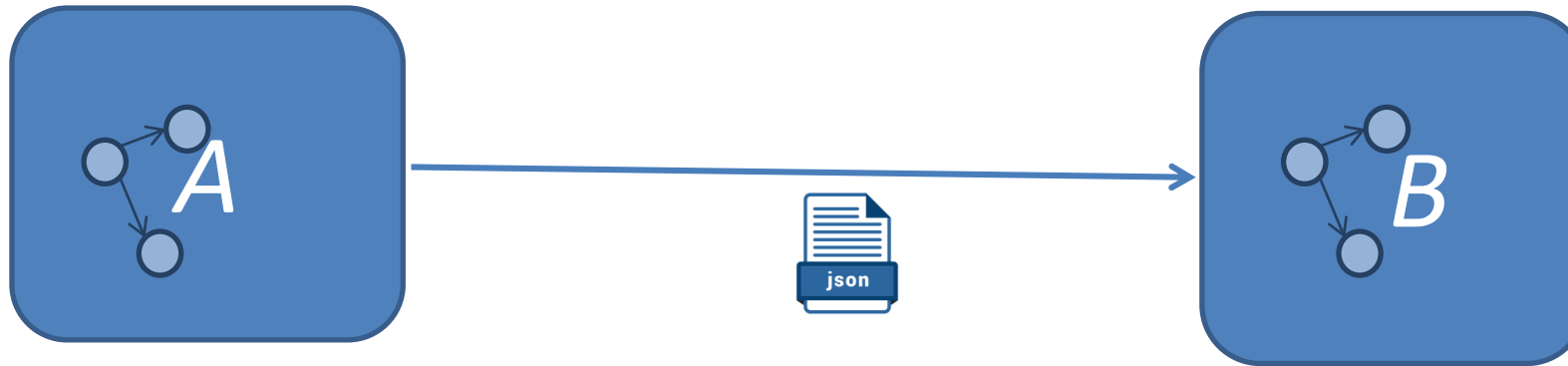


```
public class InventoryManager {  
    @Inject  
    private SystemClient systemClient;  
    ...  
}
```

Equivalent to Spring Injection
@Autowired

JSON-B & JSON-P

Open Liberty



```
public class InventoryList {  
  
    private List<SystemData> systems;  
  
    public InventoryList(List<SystemData> systems) {  
        this.systems = systems;  
    }  
  
    public List<SystemData> getSystems() {  
        return systems;  
    }  
  
    public int getTotal() {  
        return systems.size();  
    }  
}
```

```
...  
  
@GET  
@Produces(MediaType.APPLICATION_JSON)  
public InventoryList listContents() {  
    return manager.list();  
}
```




Handling 100s of collaborating and frequently evolving services requires new APIs

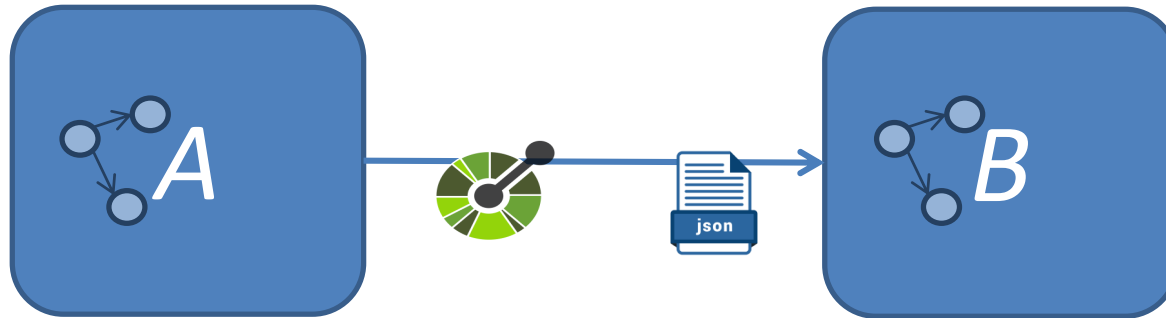
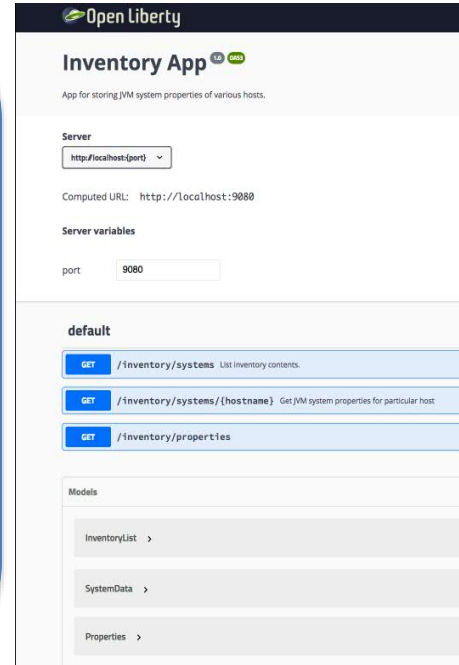
Eclipse MicroProfile

Open Liberty



MicroProfile OpenAPI

```
@GET
@Produces(MediaType.APPLICATION_JSON)
@ApiResponse(
    responseCode = "200",
    description = "host:properties pairs stored in the inventory.",
    content = @Content( mediaType = "application/json",
        schema = @Schema( type = SchemaType.OBJECT,
            implementation = InventoryList.class)))
@Operation( summary = "List inventory contents.",
    description = "Returns the stored host:properties pairs.")
public InventoryList listContents() { return manager.list(); }
```



<http://localhost:9080/openapi/ui>

openapi: 3.0.0

info:

title: Inventory App

description: App for storing JVM system properties of various hosts.

license:

name: Eclipse Public License - v 1.0

url: <https://www.eclipse.org/legal/epl-v10.html>

version: "1.0"

servers: - url: http://localhost:{port} description: Simple Open Liberty.

variables:

port:

description: Server HTTP port.

default: "9080"

paths:

/inventory/systems:

get:

summary: List inventory contents.

description: Returns the currently stored

host:properties pairs in the inventory.

operationId: listContents

responses:

200:

description: host:properties pairs stored in the inventory.

content:

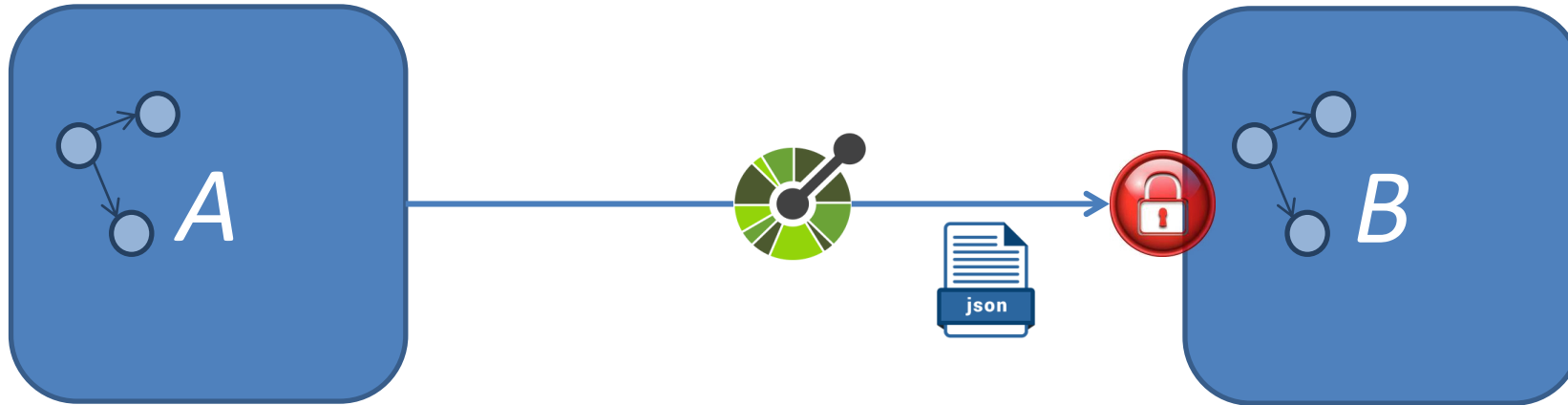
application/json:

schema:

\$ref:

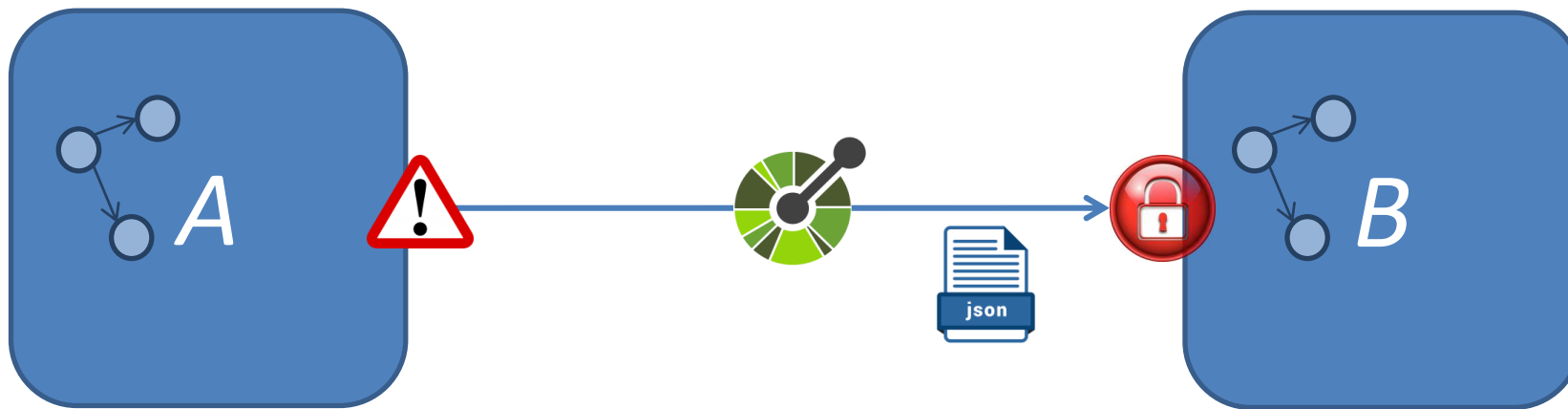
MicroProfile JWT

Open Liberty



```
@GET
@RolesAllowed({ "admin", "user" })
@Path("/{hostname}")
@Produces(MediaType.APPLICATION_JSON)
public Response getPropertiesForHost(@PathParam("hostname") String hostname,
@Context HttpHeaders httpHeaders) {...}
```

MicroProfile Fault Tolerance



```
@Fallback(fallbackMethod = "fallbackForGet")  
public Properties get(String hostname) throws IOException {  
    return invUtils.getProperties(hostname);  
}
```

MicroProfile Config

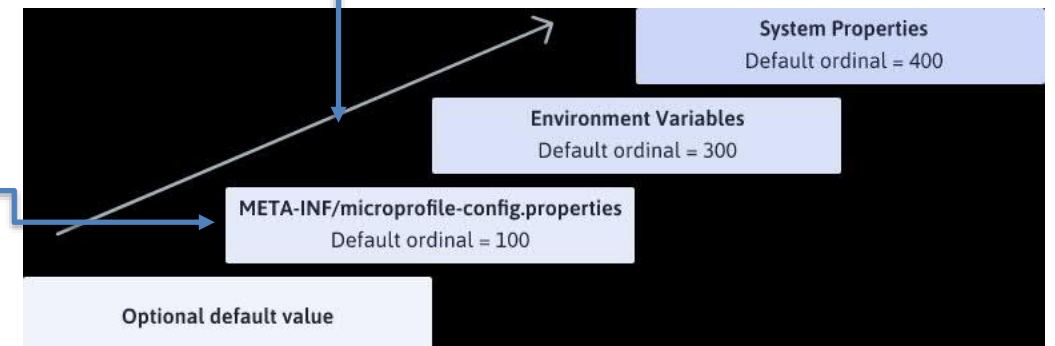
Open Liberty



```
@Inject
@ConfigProperty(name =
"io_openliberty_guides_inventory_inMaintenance")
private Provider<Boolean> inMaintenance;
```

```
config_ordinal=100
io_openliberty_guides_inventory_inMaintenance=false
```

```
{
  "config_ordinal":150,
  "io_openliberty_guides_inventory_inMaintenance":true
}
```

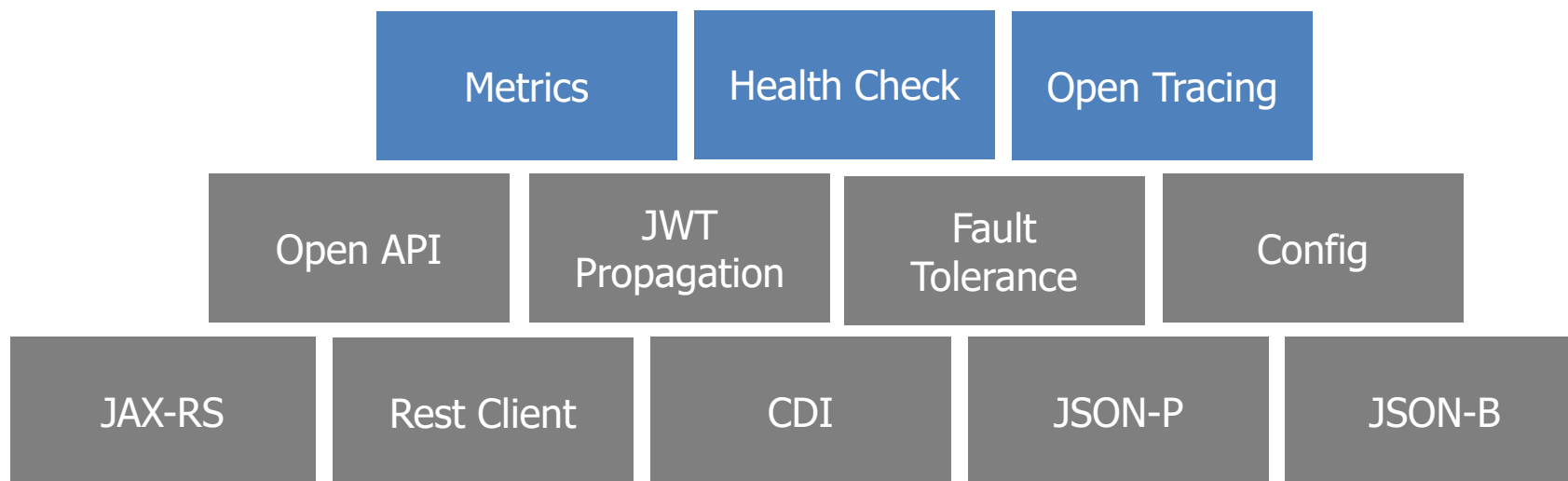
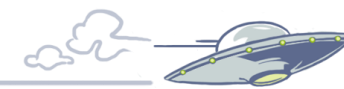




Handling 100s of collaborating services
requires a **strong operations focus**

Eclipse MicroProfile

Open Liberty

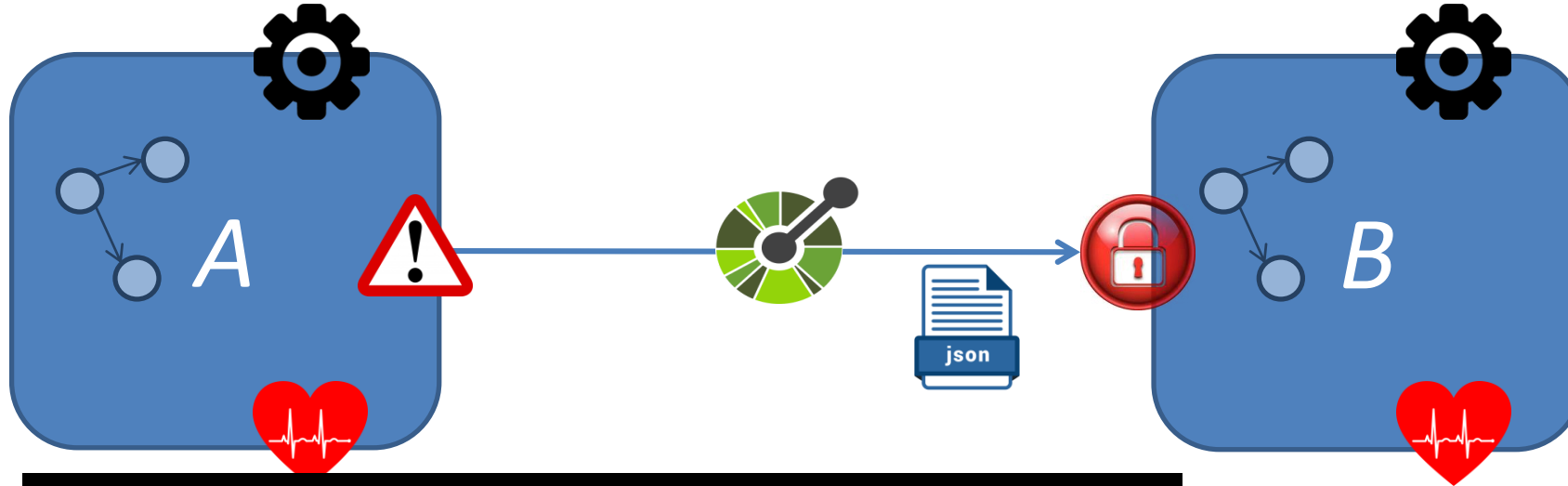


microprofile.io



MicroProfile Health

Open Liberty



```
@Health
@ApplicationScoped
public class InventoryResource implements HealthCheck {
    ...
    public boolean isHealthy() {...}

    @Override
    public HealthCheckResponse call() {
        if (!isHealthy()) {
            return HealthCheckResponse.named("InventoryResource").withData(...).down().build();
        }
        return HealthCheckResponse.named("InventoryResource").withData(...).up().build();
    }
}
```

checks:

▼ 0:

▼ data:

services:

"available"

name:

"InventoryResource"

state:

"Up"

MicroProfile Metrics

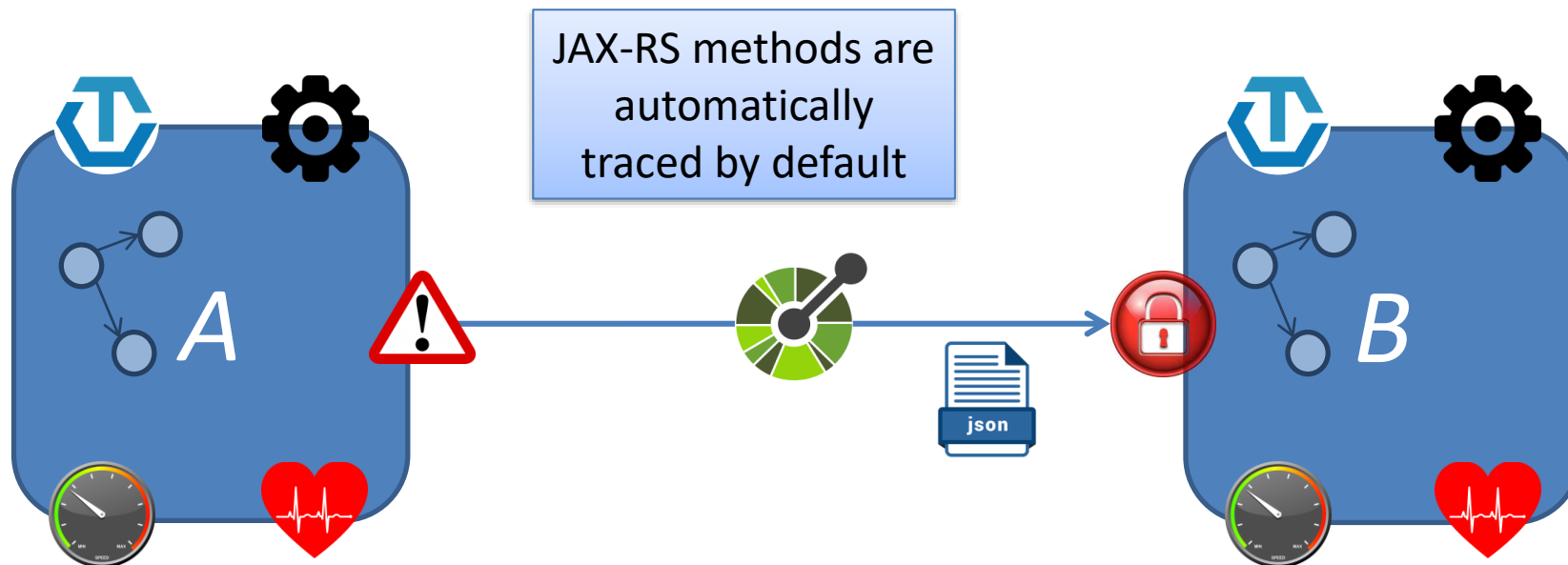
Open Liberty



```
@Timed(name = "inventoryPropertiesRequestTime",  
absolute = true,  
description = "Time needed to get the properties of" +  
"a system from the given hostname")  
public Properties get(String hostname) {  
    return invUtils.getProperties(hostname);  
}
```

MicroProfile OpenTracing

Open Liberty



```
@Traced(value = true, operationName = "InventoryManager.list")
public InventoryList list() {
    return new InventoryList(systems);
}
```



MicroProfile Starter "Beta"

Generate MicroProfile Maven Project with Examples

<https://start.microprofile.io>

Project Options

MicroProfile Server *

beans.xml *

MicroProfile Version *

Java SE Version

Examples for specifications

DOWNLOAD

⚠ "implicit" -> no beans.xml, "annotated" and "all" are the values for "bean-discovery-mode"

Guides

<https://openliberty.io/guides>

microprofile X

The quickest way to learn all things Open Liberty, and beyond!

MicroProfile - Developing microservices with ease

15 search results

4 essentials

New to MicroProfile? [Get an introduction here.](#)

Creating a RESTful web service

Learn how to create a REST service with JAX-RS, JSON-P, and Open Liberty.

🕒 30 minutes

Injecting dependencies into microservices

Learn how to use Contexts and Dependency Injection to manage and inject dependencies into microservices.

🕒 15 minutes

Consuming RESTful services with template interfaces

Learn how to use MicroProfile Rest Client to invoke RESTful services over HTTP in a type-safe way.

🕒 20 minutes

Separating configuration from code in microservices

Learn how to perform static configuration injection using MicroProfile Config.

🕒 25 minutes

+ INTERACTIVE

11 additional MicroProfile Guides

Consuming a RESTful web service

Explore how to access a simple RESTful web

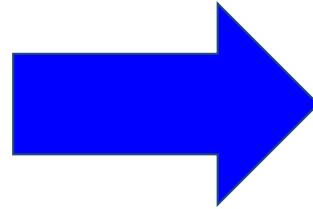
Failing fast and recovering from errors

Use MicroProfile's Timeout and Retry

Limiting the number of concurrent requests to microservices

Enabling distributed tracing in microservices

Meanwhile...



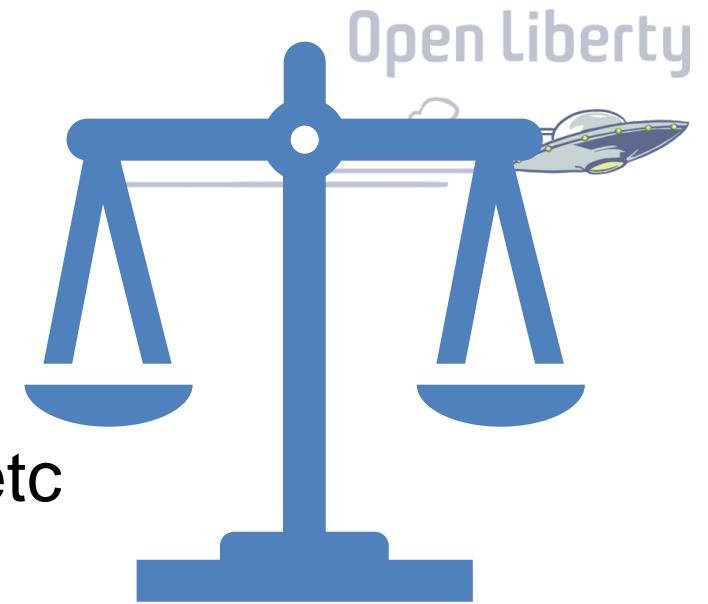
Open Liberty



JAKARTA EE

2017 - 2018

Jakarta EE vs Java EE



- Compare and Contrast
- Content, Processes, Participants, Deliverables, etc



Roadmap

Open Liberty



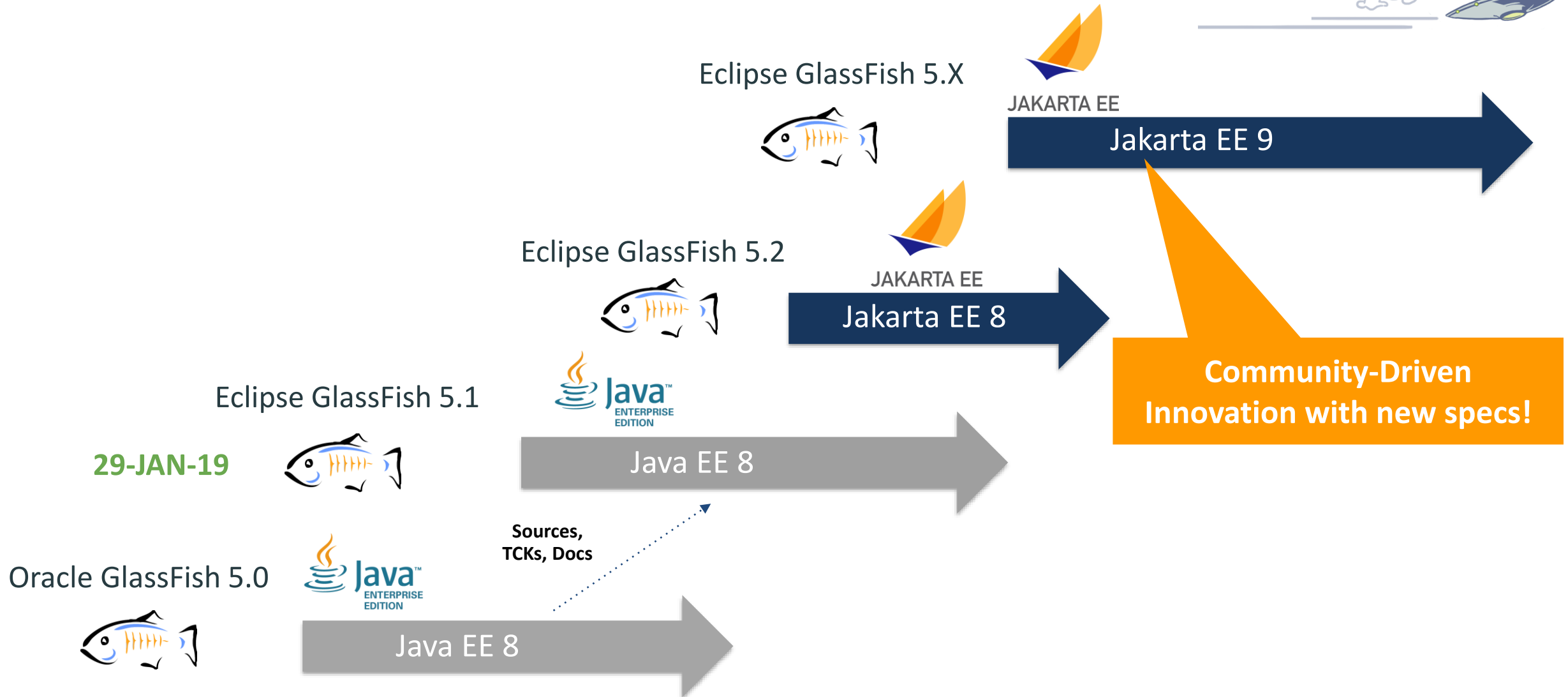
- Moving material from Oracle to Eclipse
 - Java EE 8 Specifications
 - Java EE 8 APIs
 - Java EE 8 RIs
 - Java EE 8 TCKs

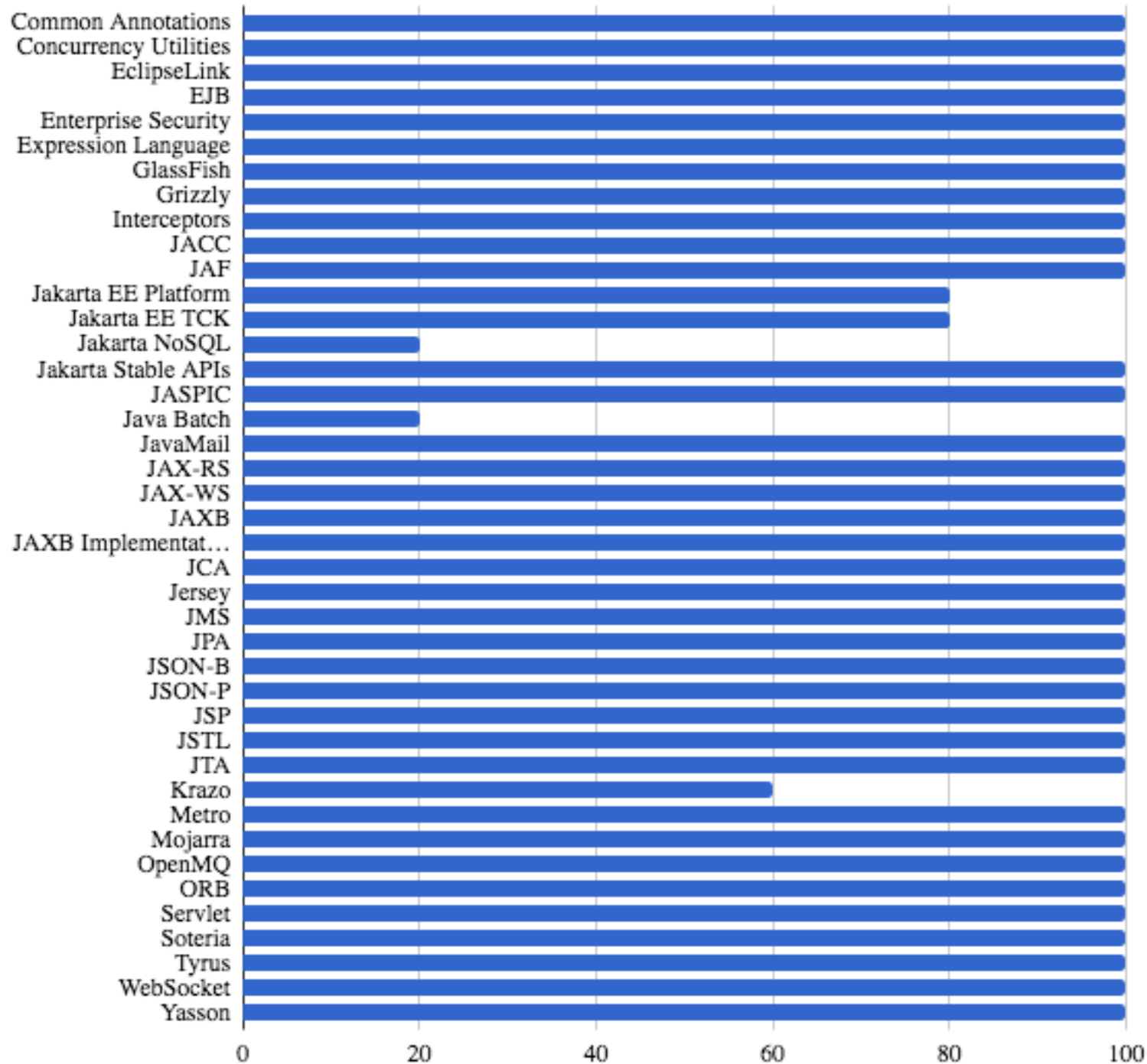


35



We are getting closer to Jakarta EE ...





Roadmap



- <https://www.eclipse.org/ee4j/status.php>
- 20% - Project Proposed
- 40% - Project Accepted and Provisioned
- 60% - Initial Code Contribution to Eclipse
- 80% - Build / Test in Github and Jenkins
- 100% - Project has First Release!

37

*As of Feb 06, 2019

Compliance Testing

Open Liberty



- Jakarta EE 8 == Java EE 8
 1. Eclipse Glassfish 5.1 is **Java EE 8 compliant** using existing Java EE 8 CTS/TCK
<http://www.oracle.com/technetwork/java/javaee/overview/compatibility-jsp-136984.html>



Java EE Compatibility

Java EE Compatibility

In every industry, businesses face the challenge of accommodating ever greater demands for high-speed data access, diverse clients, and secure transactions without incurring extensive additional costs. To extend existing IT investments while meeting these demands, developers have consistently adopted the Java Platform, Enterprise Edition.

Java EE8 Full Platform Compatible Implementations



Jan 29,
2019!

38

2. Several Compatible Implementations will be **Jakarta EE 8 compliant** using Jakarta EE 8 CTS/TCK Process

Licensing



- Oracle
 - Usage License (CDDL, GPL v2)
 - Specs, APIs, RIs
 - No TCKs available for testing!
 - Commercial License (TLDA, TCK, TM)
 - Access to TCKs for compliance testing
 - Access to Java EE brand

- Eclipse Foundation
 - Usage License (EPL v2*)
 - Specs, APIs, RIs, and TCKs!
 - <https://github.com/eclipse-ee4j/jakartaee-tck>
 - Commercial License (none)
 - Access to Jakarta EE brand (TM license)

* (Secondary) GNU General Public License, version 2 with the GNU Classpath Exception

What does it mean?

TCK is now open sourced!

Open Liberty



- **Transparency**
 - insight into tests
 - the community participation
- **Openness**
 - greater pool of contributors
 - equal opportunity with established process and governance
- **Shared burden**
 - spread responsibility for building and maintaining the TCKs
 - no dependency on a single organization or group
- **Vendor neutrality and continuity**
 - continuity in the case single entity reduces their investment.

JSR -> Eclipse Foundation Specification Process



- The JSR (for Java EE) will be replaced by the Eclipse Foundation Specification Process
 - Note: The JSR as a specification document is not going away. It still exists, but its focus is now mainly with Java SE.
 - Any specification moved to Eclipse. Once updated, the namespace needs to be updated to jakarta.* instead of using javax.*
- Eclipse Foundation Specification Process
 - Version 1.0 was defined by the Jakarta EE Specification Committee
 - <https://www.eclipse.org/projects/efsp/>
 - Covers many aspects – Specifications, APIs, Implementations, TCKs, Licensing, and Branding
 - <https://accounts.eclipse.org/mailling-list/jakarta.ee-spec>

JCP vs. EFSP

Open Liberty



Specification First



Code First



Led by Specification Lead



Collaborative



Documents and TCKs are closed source



Documents and TCKs are open source



One normative “Reference Implementation”



One or more “Compatible Implementations”



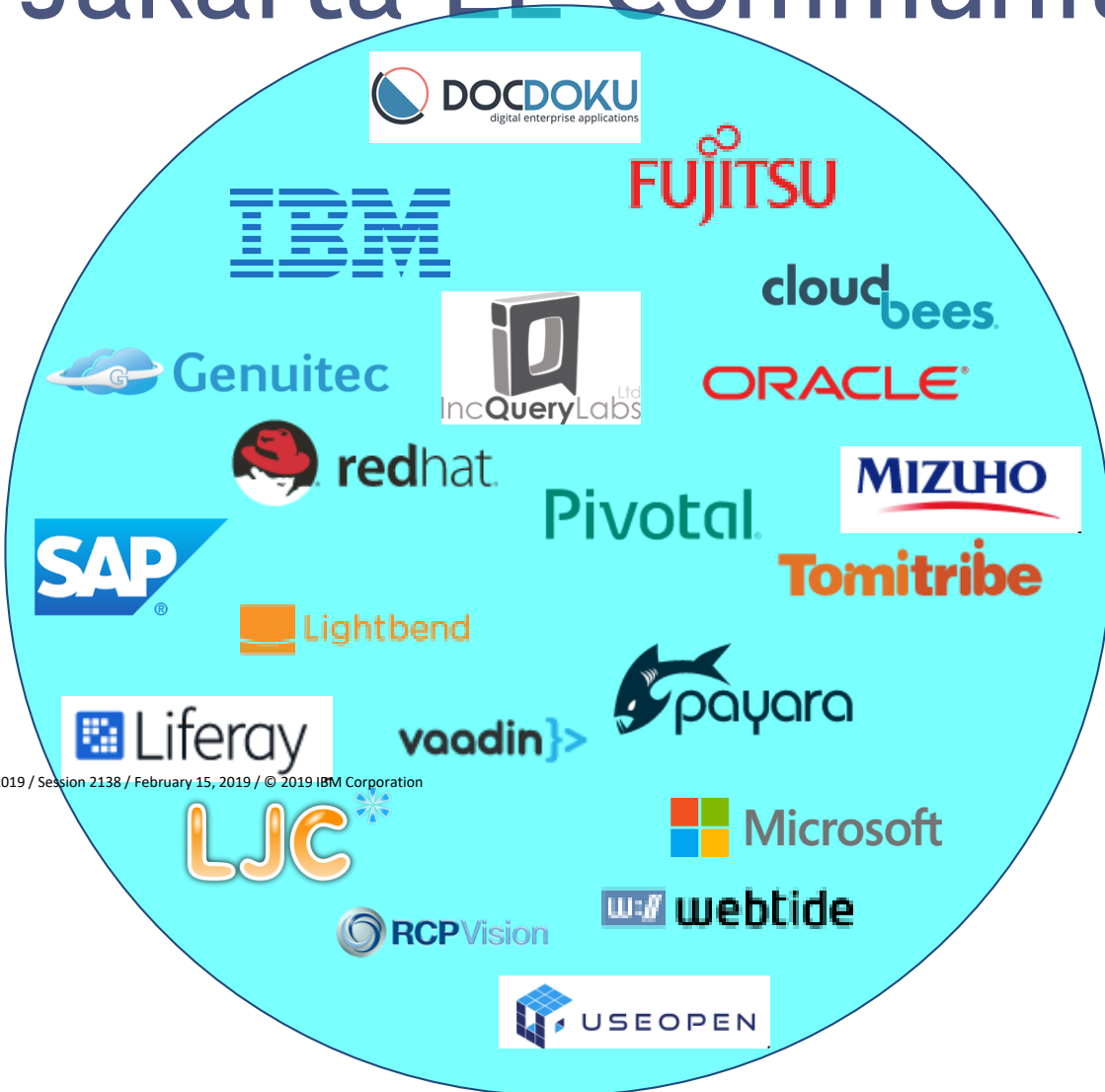
Oracle certification process



Self certification



Jakarta EE Community



Think 2019 / Session 2138 / February 15, 2019 / © 2019 IBM Corporation

43

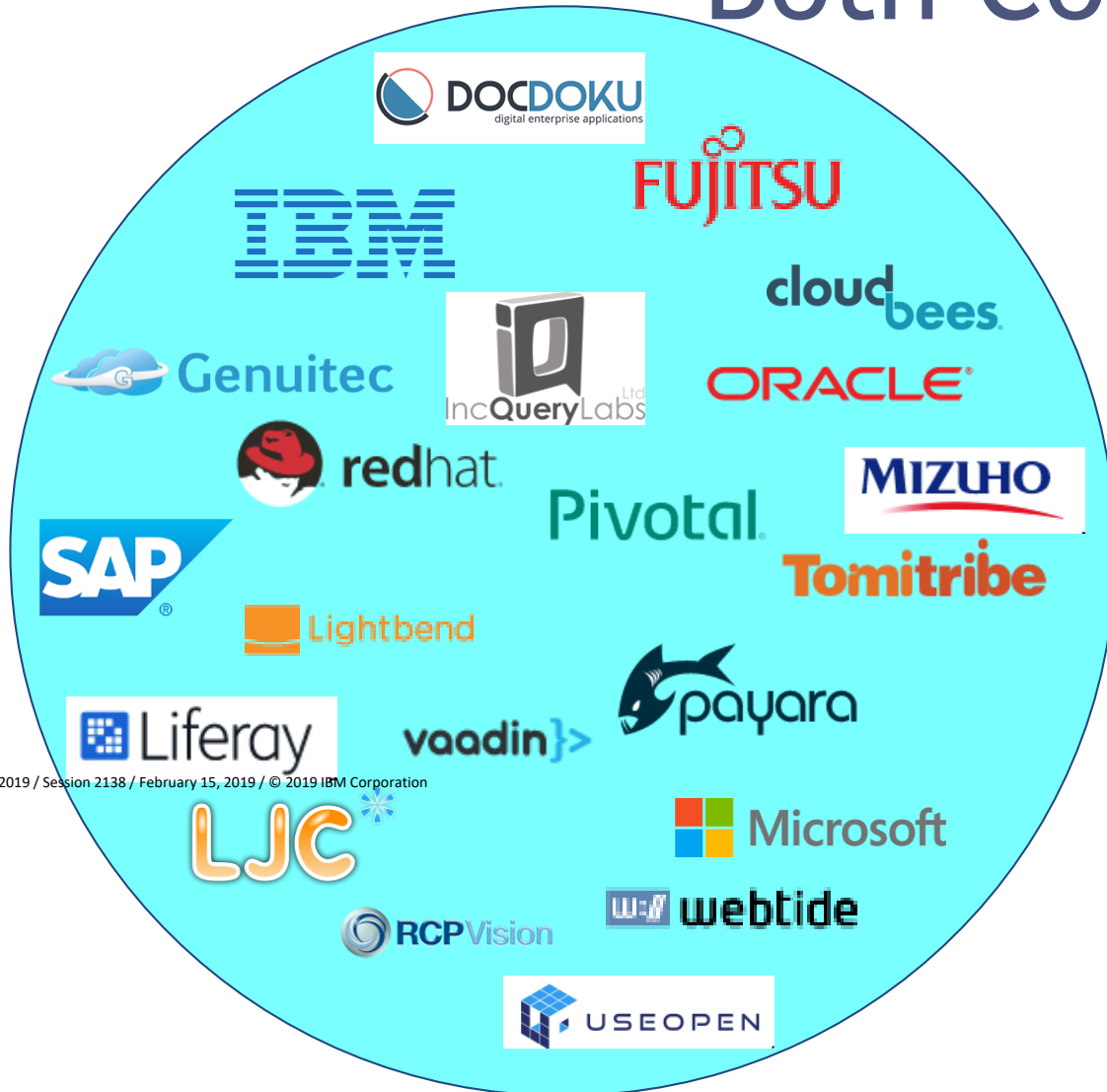
MicroProfile Community

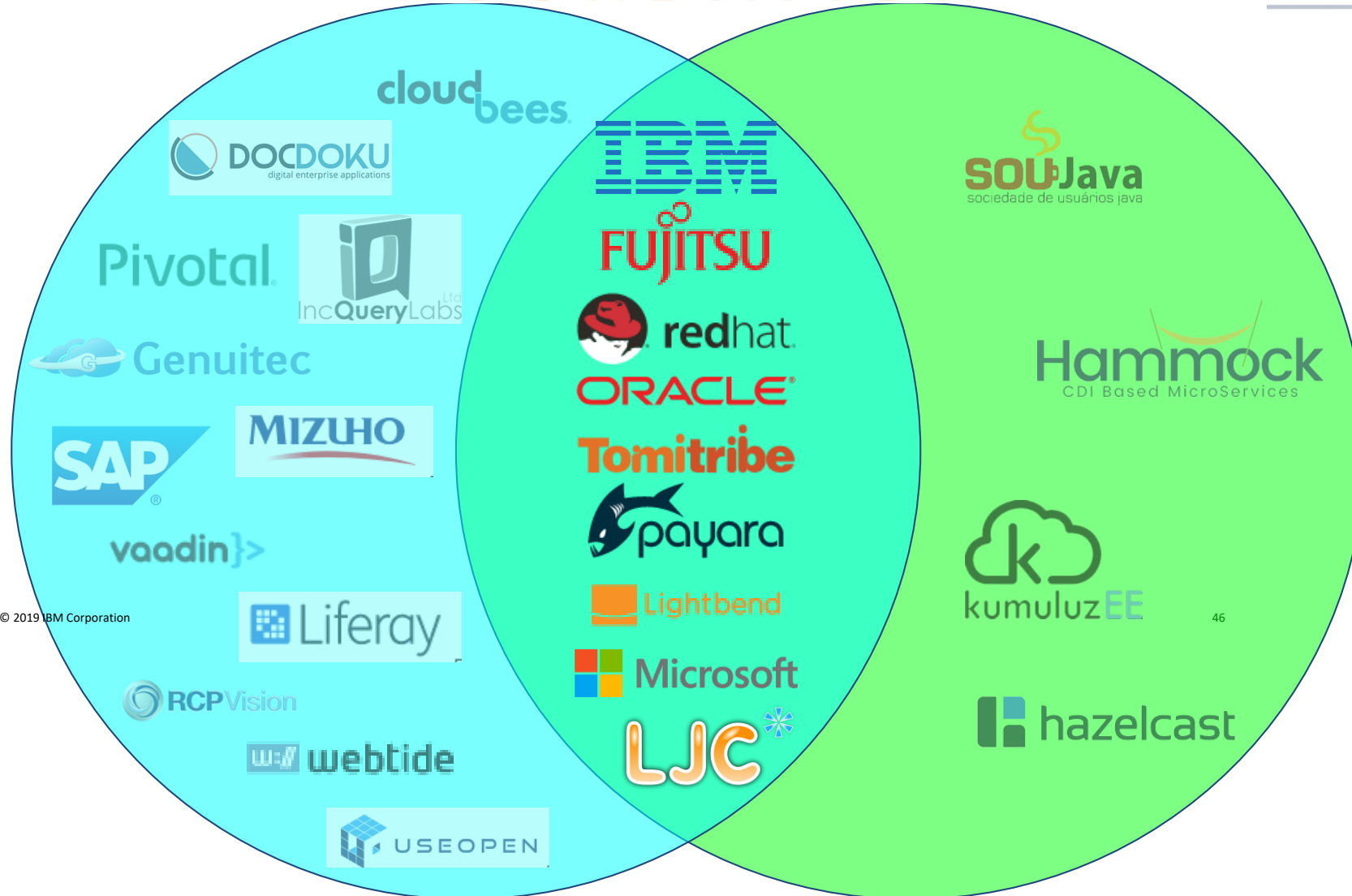
Open Liberty





Both Communities







Powerful together



JAKARTA EE



And get
involved
with





Useful Links

- <https://microprofile.io>
- <https://start.microprofile.io/>
- <https://openliberty.io/guides>
- <https://projects.eclipse.org/projects/ee4j>
- <https://github.com/eclipse-ee4j/ee4j>
- <https://github.com/eclipse-ee4j/jakartaee-platform>

极客邦科技 会议推荐2019

5月

QCon 北京

全球软件开发大会

大会: 5月6-8日
培训: 5月9-10日

QCon 广州

全球软件开发大会

培训: 5月25-26日
大会: 5月27-28日

6月

GTLC
GLOBAL
TECH LEADERSHIP
CONFERENCE

上海

技术领导力峰会

时间: 6月14-15日

GMTC 北京

全球大前端技术大会

大会: 6月20-21日
培训: 6月22-23日

7月

ArchSummit 深圳

全球架构师峰会

大会: 7月12-13日
培训: 7月14-15日

10月

QCon 上海

全球软件开发大会

大会: 10月17-19日
培训: 10月20-21日

11月

GMTC 深圳

全球大前端技术大会

大会: 11月8-9日
培训: 11月10-11日

AiCon 北京

全球人工智能与机器学习大会

大会: 11月21-22日
培训: 11月23-24日

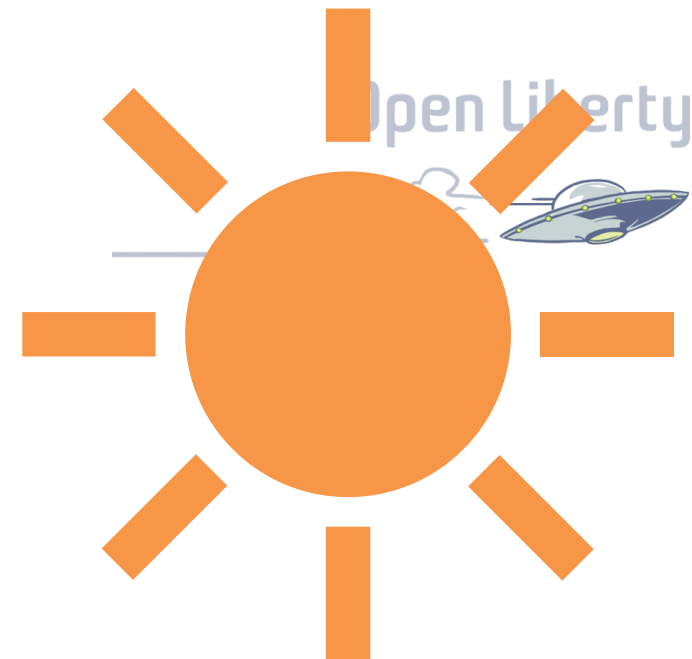
12月

ArchSummit 北京

全球架构师峰会

大会: 12月6-7日
培训: 12月8-9日

Jakarta EE 和 MicroProfile 的明天会更好！



Thank you!

