



# Red Hat Integration Bundle workshop

Nikolaj Majorov

## Table of Contents

1. History and Revisions . . . . .	1
2. Overview . . . . .	2
2.1. Need for speed: the new imperative for digital business. . . . .	2
3. Red Hat Middleware . . . . .	4
3.1. Middleware Transition . . . . .	4
3.2. Red Hat Middleware Strategy . . . . .	4
3.2.1. Red Hat Middleware Bundles . . . . .	5
3.2.2. Red Hat Integration Bundle . . . . .	6
3.3. RED HAT 3SCALE API MANAGEMENT . . . . .	7
4. Developing application with FUSE . . . . .	10
4.1. Developing API . . . . .	10
4.1.1. API First approach . . . . .	10
4.1.2. Design API with Apicurio . . . . .	10
4.2. Build and Deploy on OpenShift . . . . .	14
4.3. Code First approach . . . . .	15
4.3.1. Create Fuse7 Project for OpenShift . . . . .	15
4.3.2. Fuse Console . . . . .	16
4.3.3. CodeReady Studio . . . . .	17
5. Modernize legacy application . . . . .	19
5.1. Enterprise Service Bus monolith . . . . .	19
5.2. SOA governance / Endpoint definitions. . . . .	19
5.3. Routing . . . . .	19
5.4. Protocol transformation . . . . .	19

## 1. History and Revisions

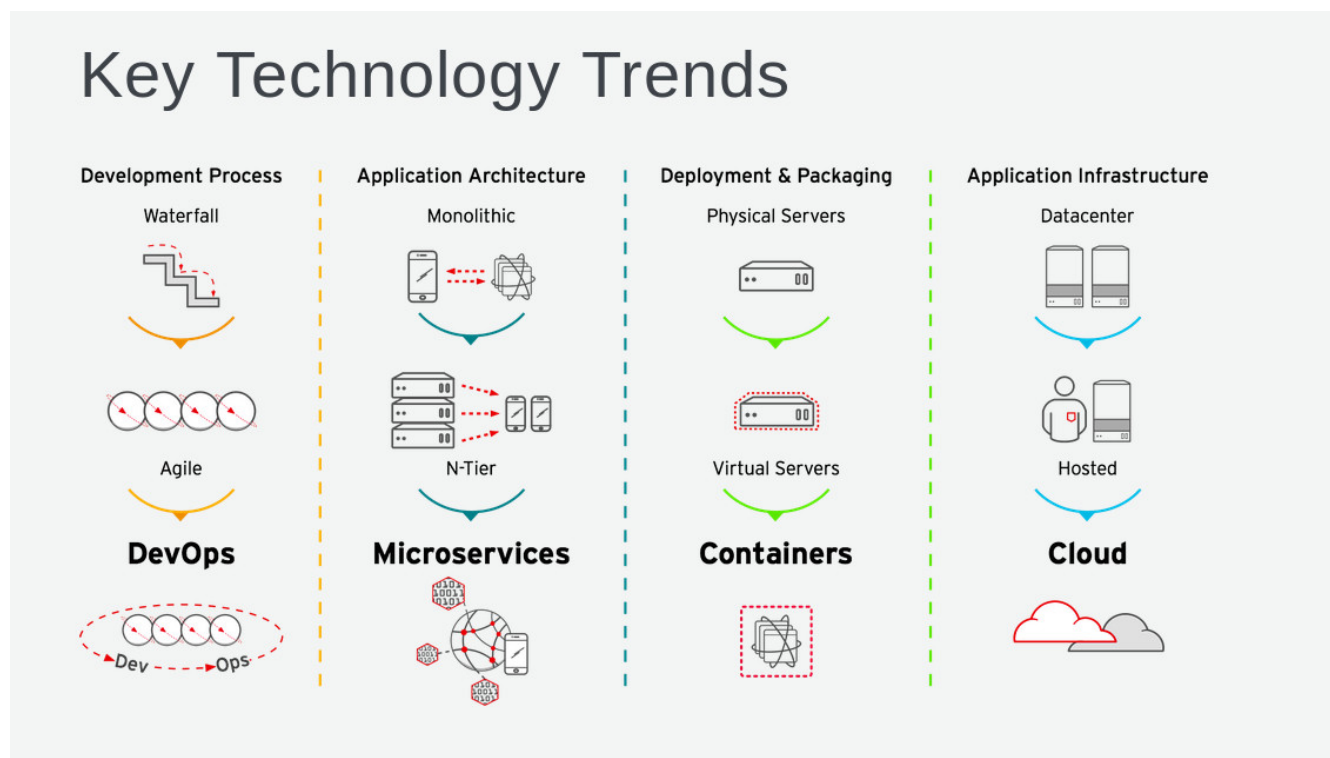
Version	Date	Authors	Changes
1.0	15.09.2019	Nikolaj Majorov <a href="mailto:nmajorov@redhat.com">nmajorov@redhat.com</a>	initial version
1.1	12.11.2019	Nikolaj Majorov <a href="mailto:nmajorov@redhat.com">nmajorov@redhat.com</a>	add fuse console

## 2. Overview

### 2.1. Need for speed: the new imperative for digital business.

With software increasingly key to how users engage with businesses and how businesses innovate to stay competitive, the speed of application development and delivery is the new digital business imperative.

Tech trends:

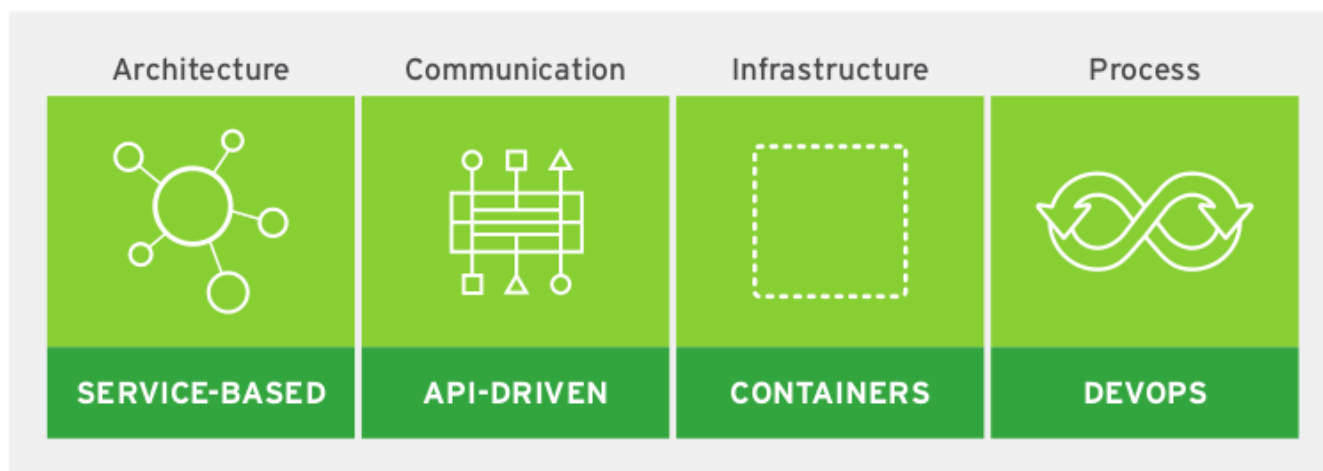


So what is Cloud Native mean ?

Cloud-native' is an adjective that describes the applications, architectures, platforms/infrastructure, and processes, that together make it economical to work in a way that allows us to improve our ability to quickly respond to change and reduce unpredictability

— CHRISTIAN POSTA, CHIEF ARCHITECT AT RED HAT AND AUTHOR OF MICROSERVICES FOR JAVA DEVELOPERS

Cloud Native Application characteristics:



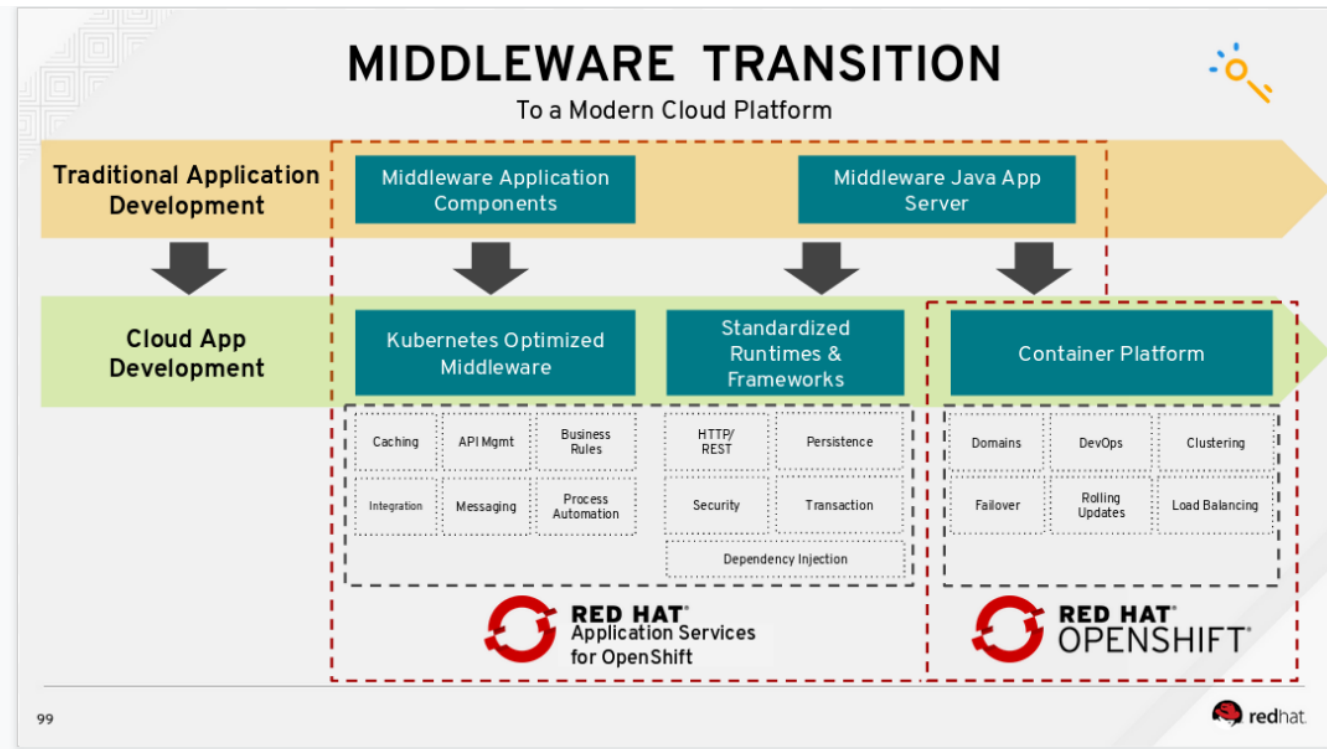
Tools are also important:

We want to make sure that all the engineers at Netflix are able to work as fast and as responsibly fast as they want to work. If we don't give them enough context about what they're deploying or about what's going on in the ecosystem at large then they could cause big problems in the service. They could cause disruptions in the service.

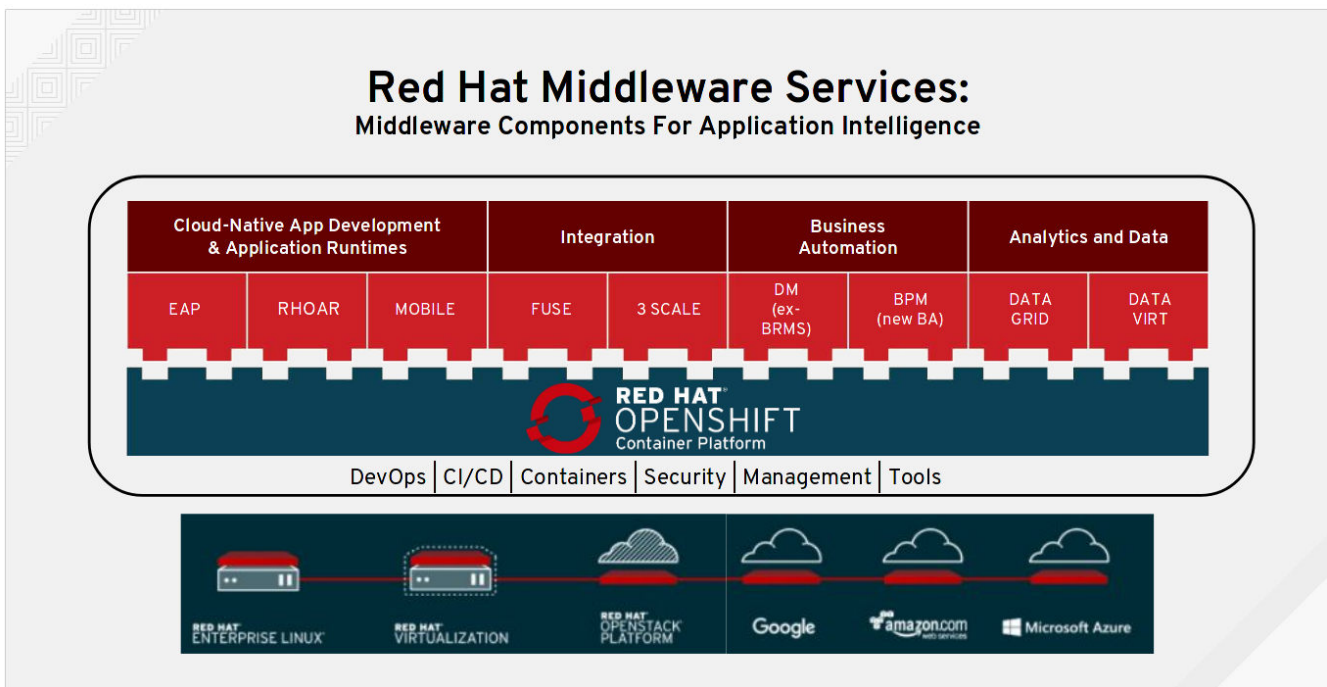
— Dianne Marsh, director of engineering for engineering tools at Netflix.

3. Red Hat Middleware

3.1. Middleware Transition



3.2. Red Hat Middleware Strategy



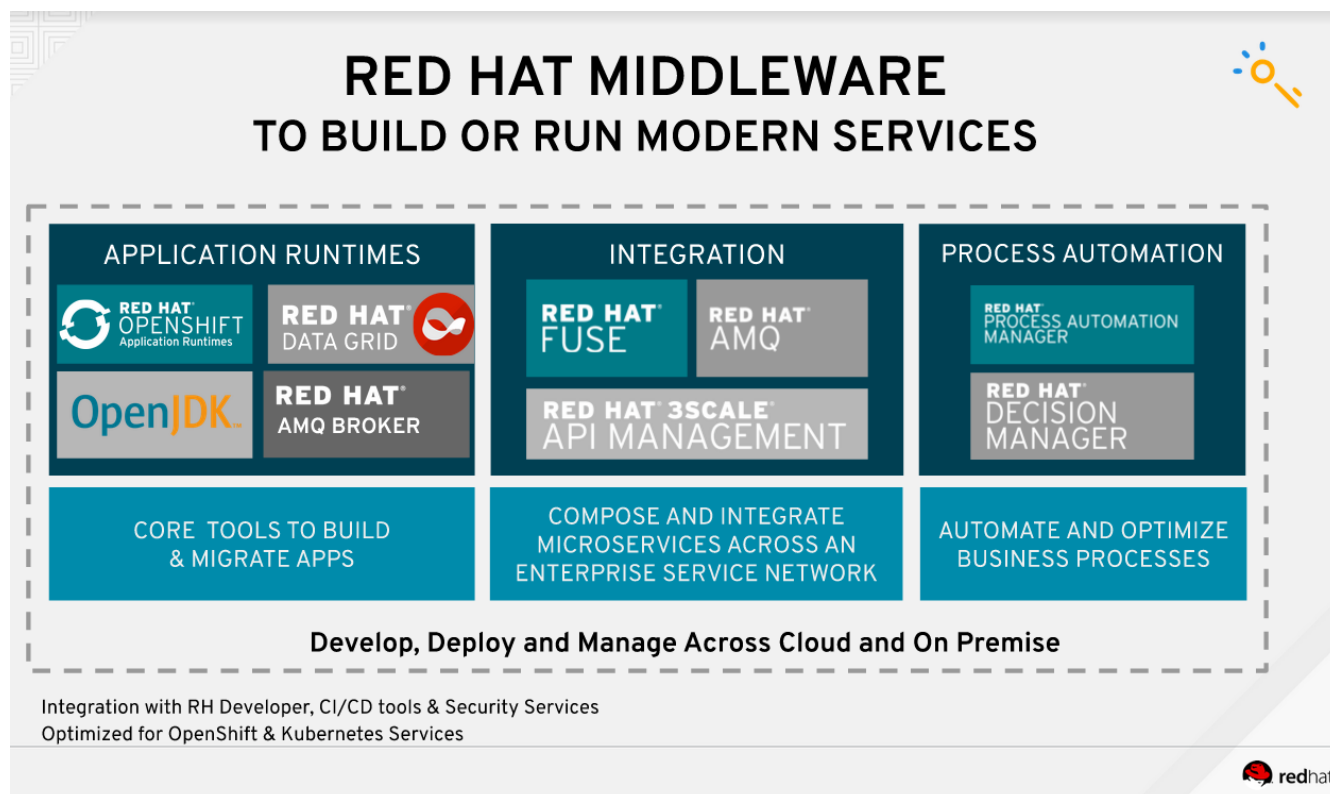
### 3.2.1. Red Hat Middleware Bundles

Red Hat offers bundled subscriptions . They can be deployed on-premise or to a public cloud - with or without OpenShift.

Subscription Name	Red Hat Supported Products and Components Included
Red Hat Application Runtimes	JBoss EAP
	JBoss Web Server
	OpenShift Application Runtimes
	Red Hat Data Grid
	AMQ (broker only)
	Red Hat Single Sign-On
	OpenJDK
	JBoss Core Services
Red Hat Integration	All of the products included in Application Runtimes
	Fuse
	AMQ (full product with AMQ Streams)
	3scale API Management

Overview of RedHat bundles can be found here:

<https://access.redhat.com/articles/3666991>



**RED HAT MIDDLEWARE  
TO BUILD OR RUN MODERN SERVICES**

The diagram illustrates the Red Hat Middleware ecosystem for building or running modern services, organized into three main functional areas:

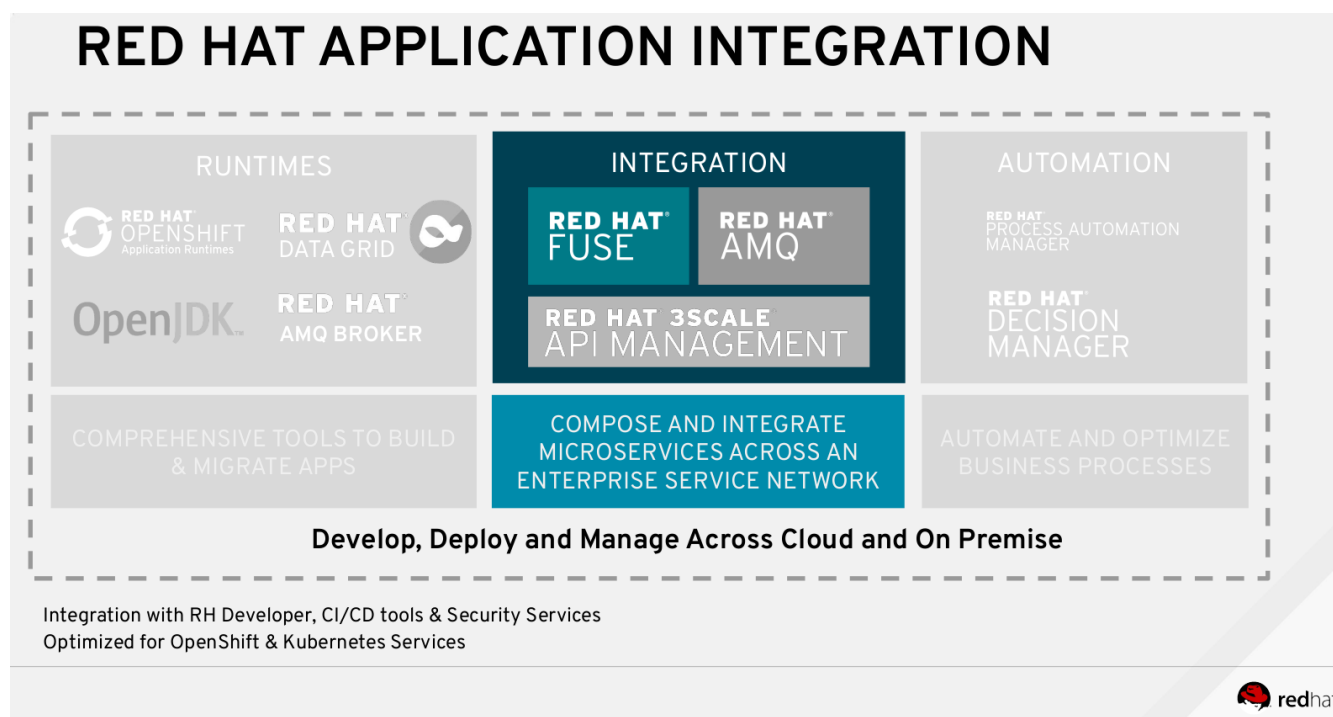
- APPLICATION RUNTIMES:** Includes Red Hat OpenShift Application Runtimes, Red Hat Data Grid, OpenJDK, and Red Hat AMQ Broker. This area focuses on **CORE TOOLS TO BUILD & MIGRATE APPS**.
- INTEGRATION:** Includes Red Hat Fuse, Red Hat AMQ, and Red Hat 3Scale API Management. This area focuses on **COMPOSE AND INTEGRATE MICROSERVICES ACROSS AN ENTERPRISE SERVICE NETWORK**.
- PROCESS AUTOMATION:** Includes Red Hat Process Automation Manager and Red Hat Decision Manager. This area focuses on **AUTOMATE AND OPTIMIZE BUSINESS PROCESSES**.

These components are used to **Develop, Deploy and Manage Across Cloud and On Premise**.

Integration with RH Developer, CI/CD tools & Security Services  
Optimized for OpenShift & Kubernetes Services

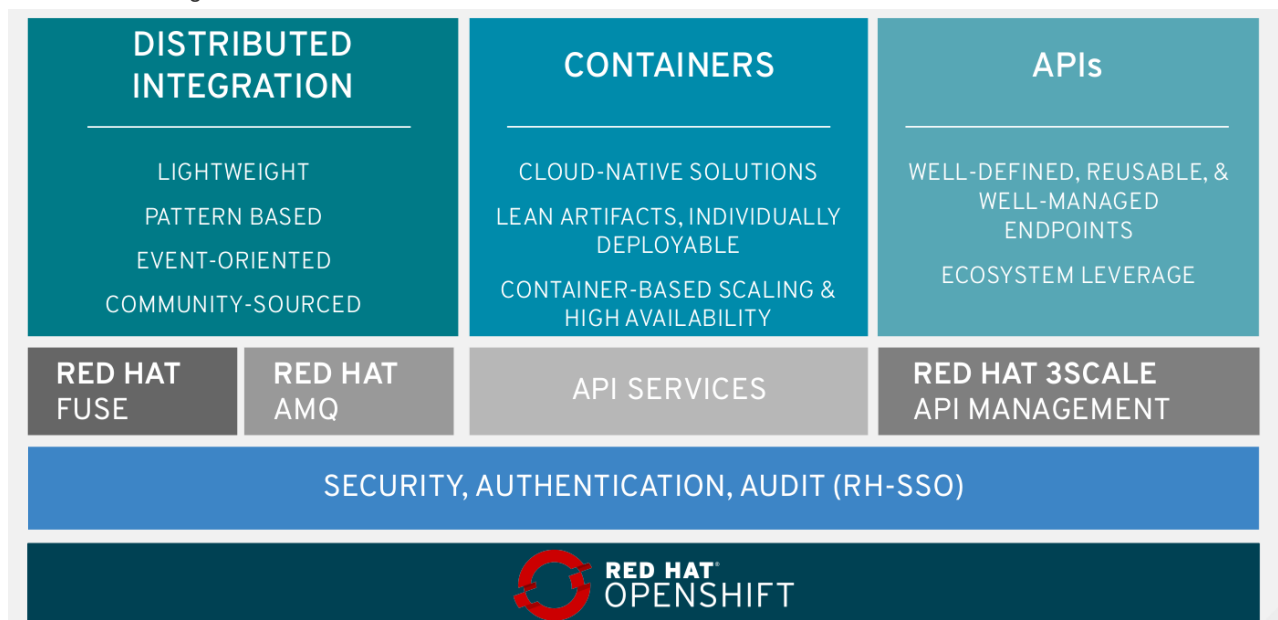
redhat

## 3.2.2. Red Hat Integration Bundle



All of the products included in Application Runtimes plus:

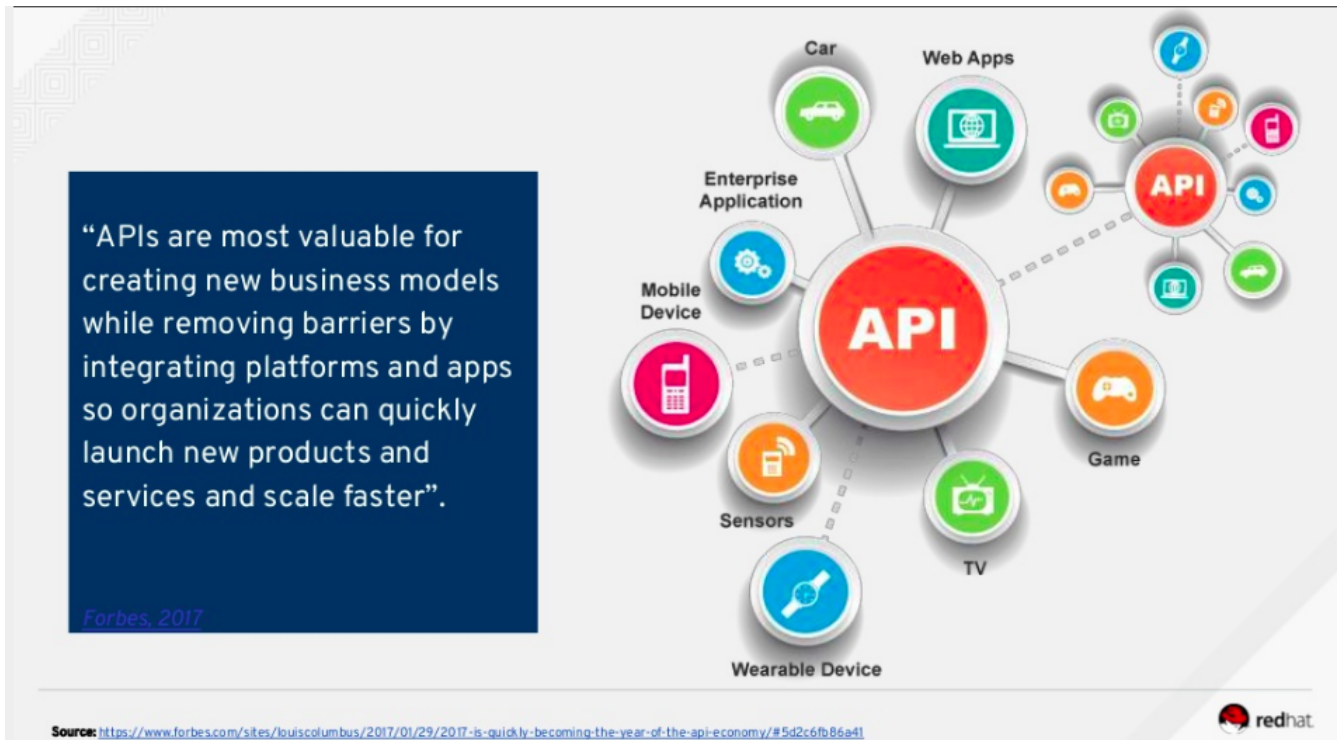
- Fuse
- AMQ (full product with AMQ Streams)
- 3scale API Management



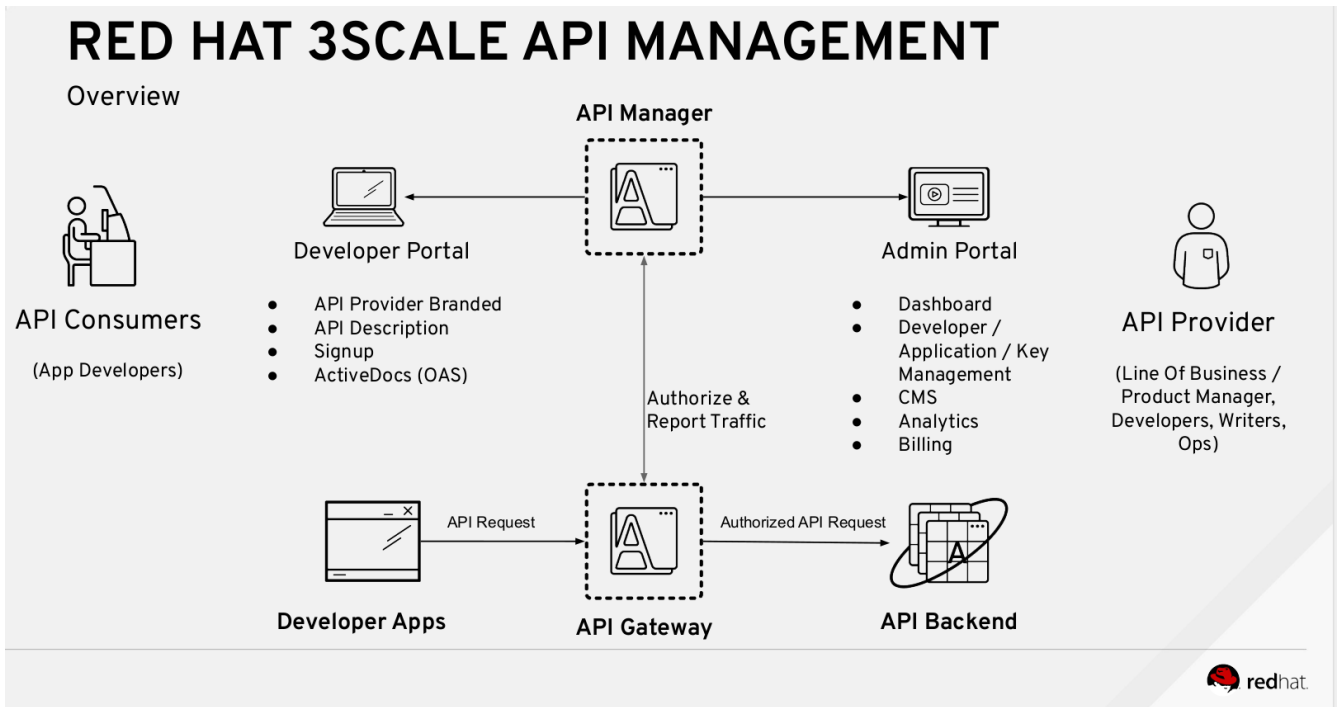


### 3.3. RED HAT 3SCALE API MANAGEMENT

API why so important ?



3scale Overview



Features:

# RED HAT 3SCALE API MANAGEMENT

API FIRST FOR Cloud MICROSERVICES

Control	Visibility	Flexibility
<ul style="list-style-type: none"> <li>• Security</li> <li>• Key management</li> <li>• Rate limiting</li> <li>• Policy enforcement</li> <li>• App and user management</li> <li>• Provisioning</li> </ul>	<ul style="list-style-type: none"> <li>• Analytics</li> <li>• App tracking</li> <li>• User tracking</li> <li>• Traffic alerts</li> <li>• Engagement</li> <li>• Developer support</li> </ul>	<ul style="list-style-type: none"> <li>• Distributed</li> <li>• Multi-department</li> <li>• Multi-environment</li> <li>• Highly scalable</li> <li>• Powerful APIs</li> <li>• Webhooks</li> </ul>



## Features:

- Scalability and Uptime

The API management platform you choose shouldn't be a bottleneck or a single point of failure. It must provide caching, fault tolerance, traffic routing, and load balancing. 3scale is designed to scale to billions of API calls

- Developer Portal

A developer portal is key to ensuring a positive experience for developers. To keep them engaged, provide the tools they need to get started in minutes with use cases, example code, documentation, and pricing.

- API Access Control and Security

3scale's powerful API access, policy and traffic controls make it simple to authenticate traffic, restrict by policy, protect backend services, impose rate limits and create access tiers. Provide sophisticated support for authentication with API keys, OIDC tokens or custom configs. Classify different types of users and provide a variety of business services. Application plans help control what can be done with your API. Rate limits allow you to manage and control flow.

- Analytics

You need insight into how your APIs are performing: access to trends, peak usage times, which applications generate the most traffic, which APIs are most popular, and which APIs or endpoints are used the least.

- Monetization

APIs can enable whole new revenue streams so the ability to set pricing rules, invoicing, and payment collection is critical.

Full description of Supported configuration can be found here:

<https://access.redhat.com/articles/2798521>

But enterprises are learning that it's important for API programs to consider more than gateways; analysts Kevin Matheny and Matt Braiser note in the Gartner report "[How to Successfully Implement API Management](#),"

"API management requires an API gateway for endpoint protection, but a gateway alone is insufficient. Gateways do not offer analytics, monitoring, developer support or governance capabilities."

### 4. Developing application with FUSE

Before you start, make sure you have installed the following software:

[SE Development Kit \(JDK\) version 1.8.x \(Java 8\)](#)

[Apache Maven](#)

This workshop is for Red Hat integration bundle and shows simple case for developing an a API based Cloud Native application.

#### 4.1. Developing API

##### 4.1.1. API First approach

An API-first approach means that for any given development project, your APIs are treated as “first-class citizens.” That everything about a project revolves around the idea that the end product will be consumed by mobile devices, and that APIs will be consumed by client applications.

<https://swagger.io/resources/articles/adopting-an-api-first-approach/>

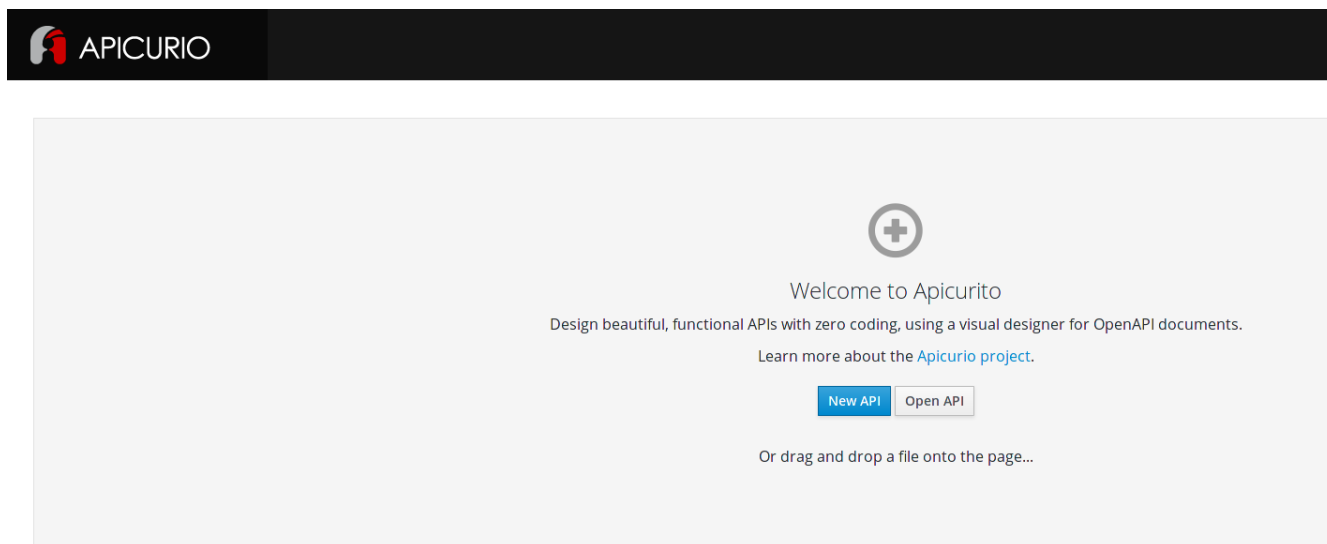
The Benefits of an API-First Approach:

- Development teams can work in parallel
- Ensures good developer experiences
- Reduces the risk of failure

##### 4.1.2. Design API with Apicurio

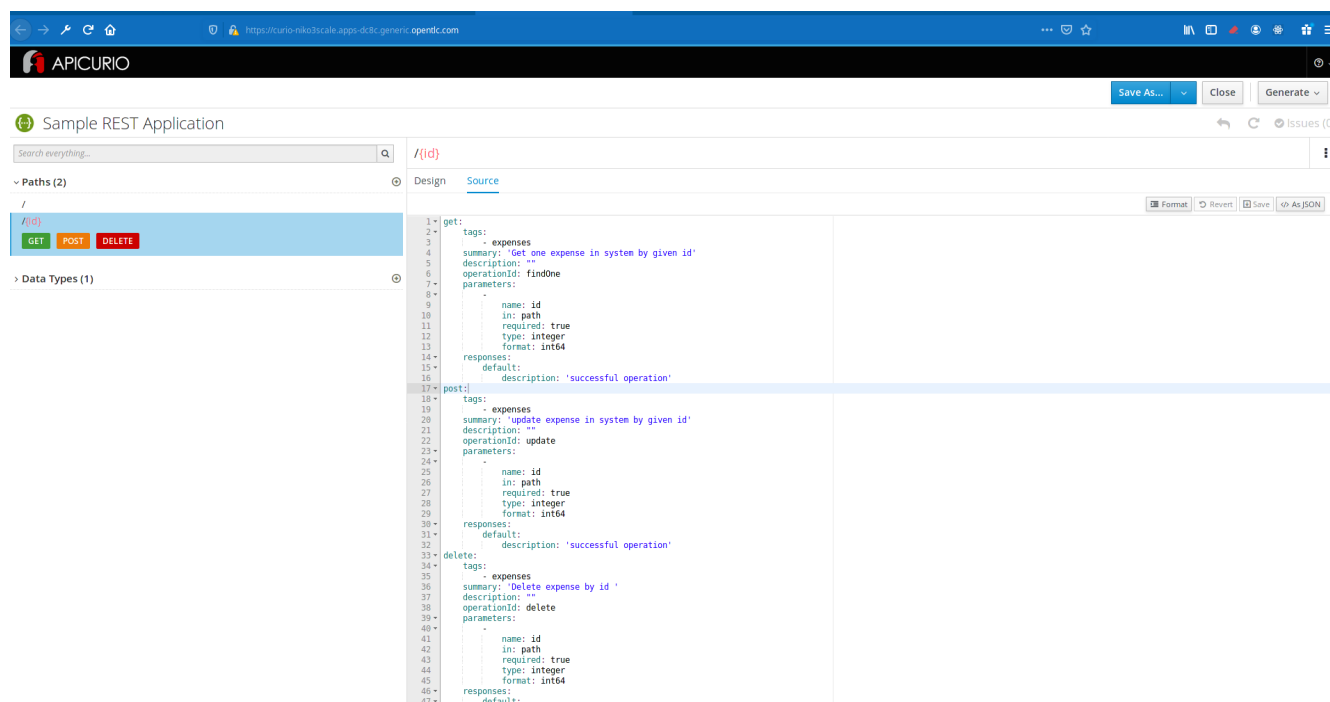
The officially supported is a [apicurito](#) and it is installed with operator



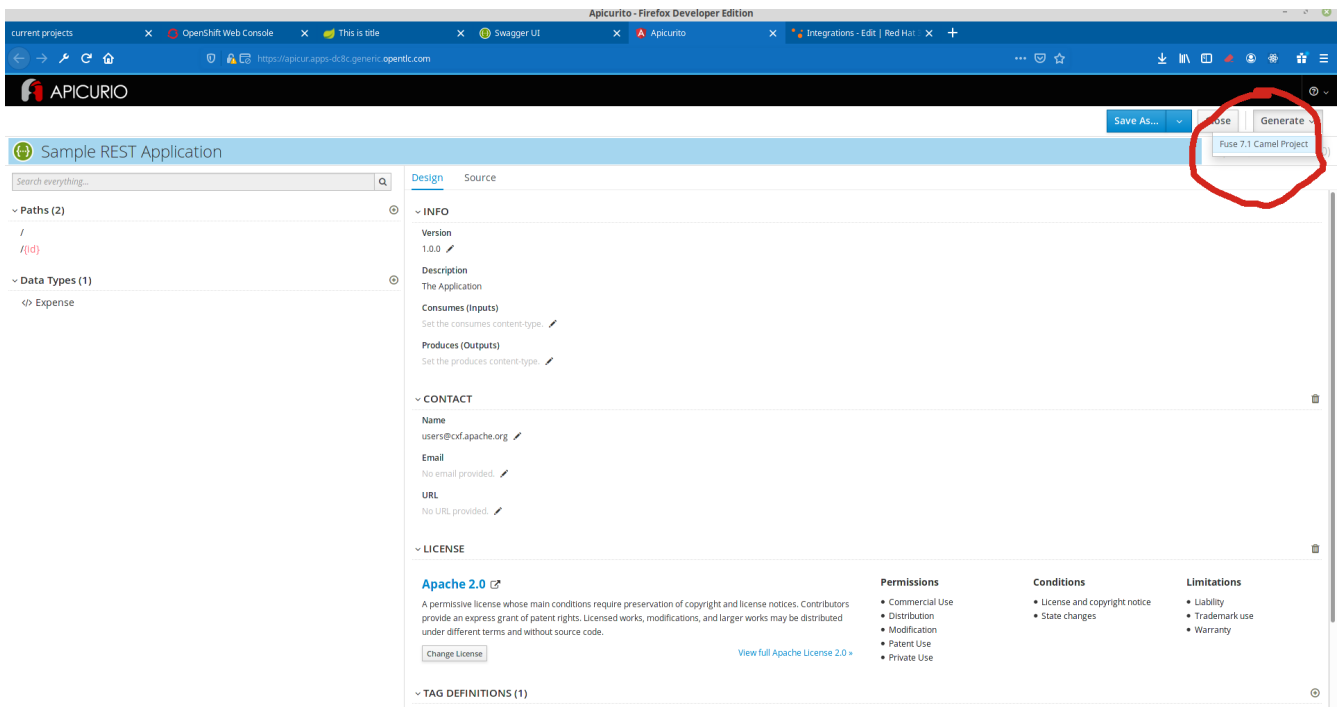


Official opensource project is [Apicurio Studio](#) and it can generate [quarkus](#) framework

Design your API with [OpenAPI v3.0](#):



Click generate button from Apicurio gui. It will generate you a fuse7 project - download it unzip, correct and run.



Another option will be to export openapi json file and generate java classes using swagger maven plugin.

Example for pom.xml with swagger-maven plugin configuration:

```
<plugin>
  <groupId>io.swagger.codegen.v3</groupId>
  <artifactId>swagger-codegen-maven-plugin</artifactId>
  <version>3.0.11</version>
  <configuration>
    <inputSpec>${swagger.file}</inputSpec>
  </configuration>
  <executions>
    <execution>
      <id>generate-swagger-spring</id>
      <phase>generate-sources</phase>
      <goals>
        <goal>generate</goal>
      </goals>
      <configuration>
        <language>spring</language>
        <modelPackage>${project.groupId}.swagger.model</modelPackage>
        <apiPackage>${project.groupId}.swagger.api</apiPackage>
        <invokerPackage>${project.groupId}.swagger.invoker</invokerPackage>
      </configuration>
    </execution>
  </executions>
</plugin>
```

Example for project can be found in repository:

fuse7-api-first/service-generated

## 4.2. Build and Deploy on OpenShift

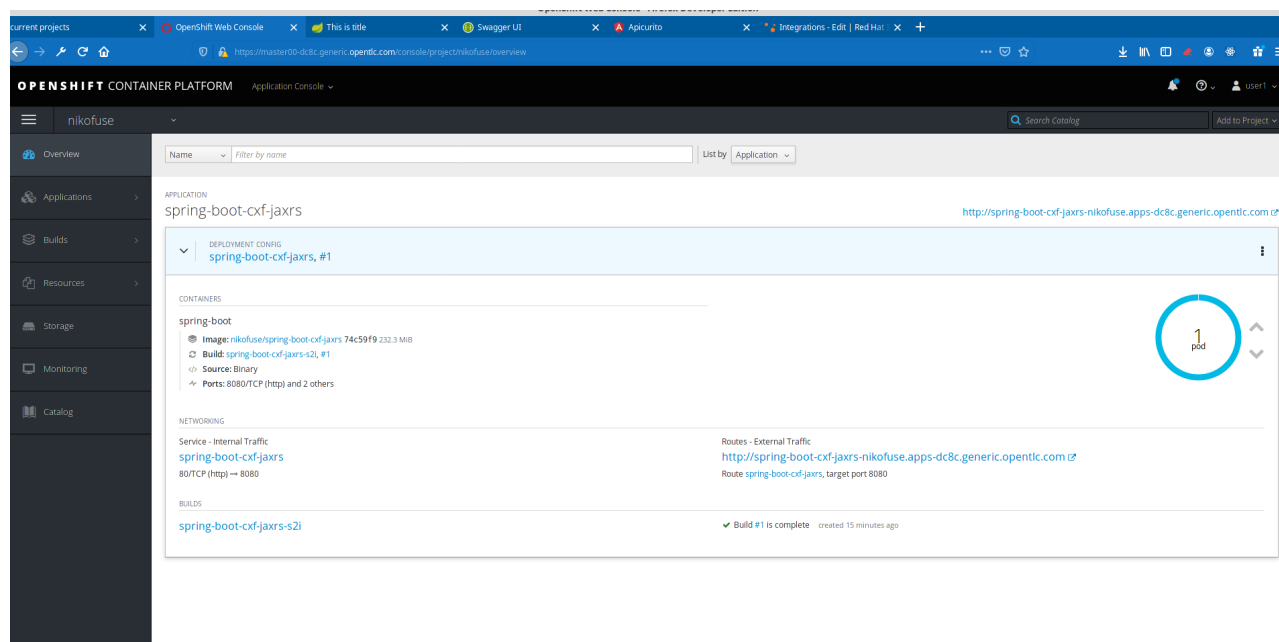
login to your OpenShift and select project:

```
oc project
Using project "nikofuse" on server "https://master00-dc8c.generic.opentlc.com:443".
mvn clean -DskipTests fabric8:deploy -Popenshift

#some seconds later...

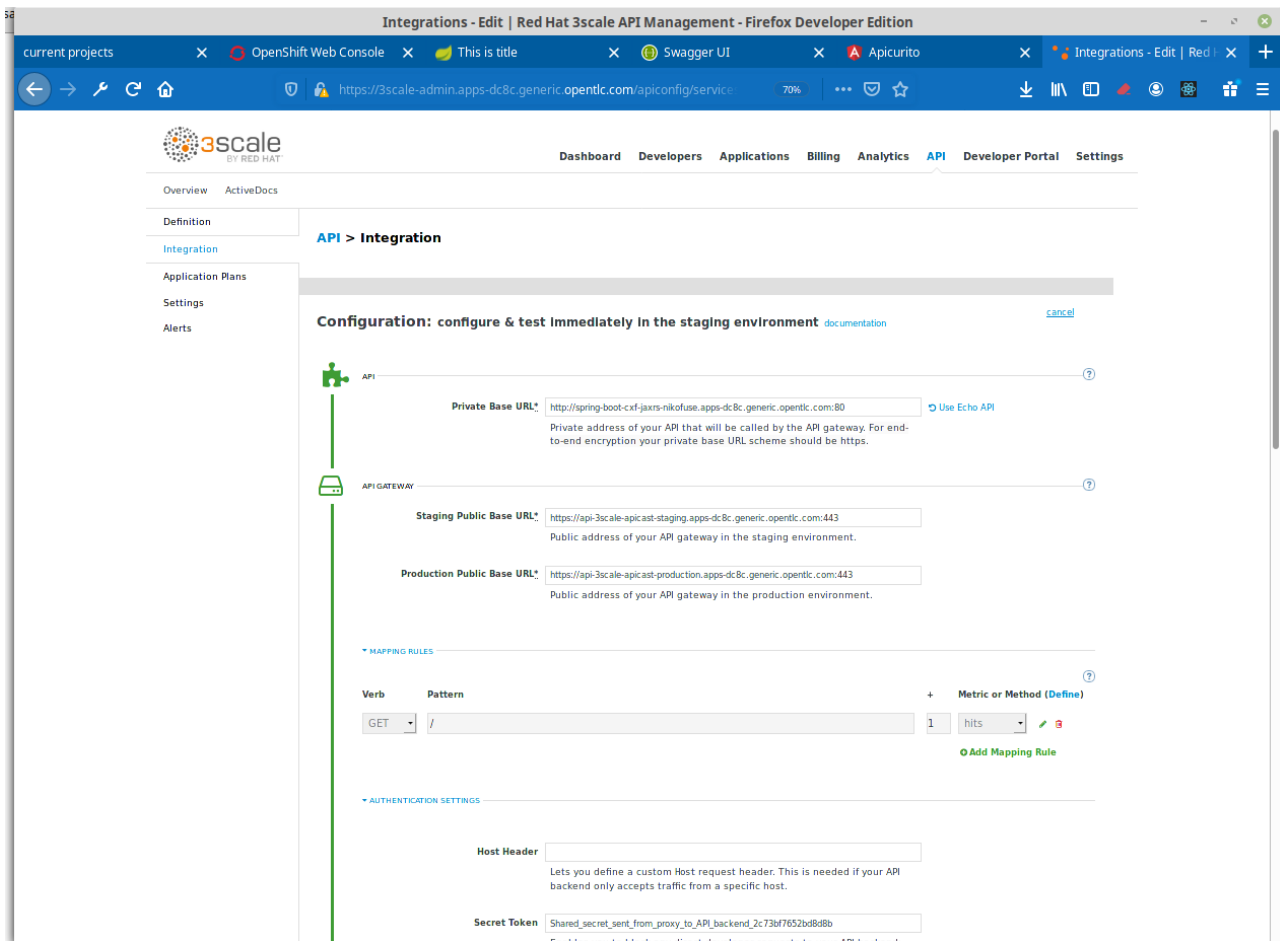
[INFO] Creating a Service from openshift.yml namespace nikofuse name spring-boot-cxf-jaxrs
[INFO] Created Service: target/fabric8/applyJson/nikofuse/service-spring-boot-cxf-jaxrs.json
[INFO] Using project: nikofuse
[INFO] Creating a DeploymentConfig from openshift.yml namespace nikofuse name spring-boot-cxf-jaxrs
[INFO] Created DeploymentConfig: target/fabric8/applyJson/nikofuse/deploymentconfig-spring-boot-cxf-jaxrs.json
[INFO] Creating Route nikofuse:spring-boot-cxf-jaxrs host: null
[INFO] F8: HINT: Use the command 'oc get pods -w' to watch your pods start up
[INFO] -----
[INFO] BUILD SUCCESS
[INFO] -----
[INFO] Total time: 05:54 min
[INFO] Finished at: 2019-09-19T10:56:45+02:00
[INFO] -----
```

See deployment in the OpenShift web-console:



Add api to 3scale:





Enhance api security with key !

and call it:

```
curl -kv "https://api-3scale-apicast-staging.apps-dc8c.generic.opentlc.com:443/services/helloservice?user_key=3d4094d3eb6c056e455bfdccd6f010c5"
```

You create your first secure api !

## 4.3. Code First approach

### 4.3.1. Create Fuse7 Project for OpenShift

```
mvn org.apache.maven.plugins:maven-archetype-plugin:2.4:generate \
  -DarchetypeCatalog=https://maven.repository.redhat.com/ga/io/fabric8/archetypes/archetypes-catalog/2.2.0.fuse-740017-redhat-00003/archetypes-catalog-2.2.0.fuse-740017-redhat-00003-archetype-catalog.xml \
  -DarchetypeGroupId=org.jboss.fuse.fis.archetypes \
  -DarchetypeArtifactId=spring-boot-camel-xml-archetype \
  -DarchetypeVersion=2.2.0.fuse-740017-redhat-00003
```

to generate Swagger support

```

mvn org.apache.maven.plugins:maven-archetype-plugin:2.4:generate \
  -DarchetypeCatalog=https://maven.repository.redhat.com/ga/io/fabric8/archetypes/archetypes-catalog/2.2.0.fuse-740017-redhat-00003/archetypes-catalog-2.2.0.fuse-740017-redhat-00003-archetype-catalog.xml \
  -DarchetypeGroupId=org.jboss.fuse.fis.archetypes \
  -DarchetypeArtifactId=spring-boot-cxf-jaxrs-archetype \
  -DarchetypeVersion=2.2.0.fuse-740017-redhat-00003

```

....

```

Define value for property 'artifactId': : epenxes-manager
Define value for property 'version': : 1.0-SNAPSHOT: :
Define value for property 'package': : fuse.redcloud.site: :
Confirm properties configuration:
groupId: redcloud.site
artifactId: expenses-manager
version: 1.0-SNAPSHOT
package: fuse.redcloud.site
Y: : y

```

### 4.3.2. Fuse Console

Fuse console used to enable view on all running applications in the namespace.

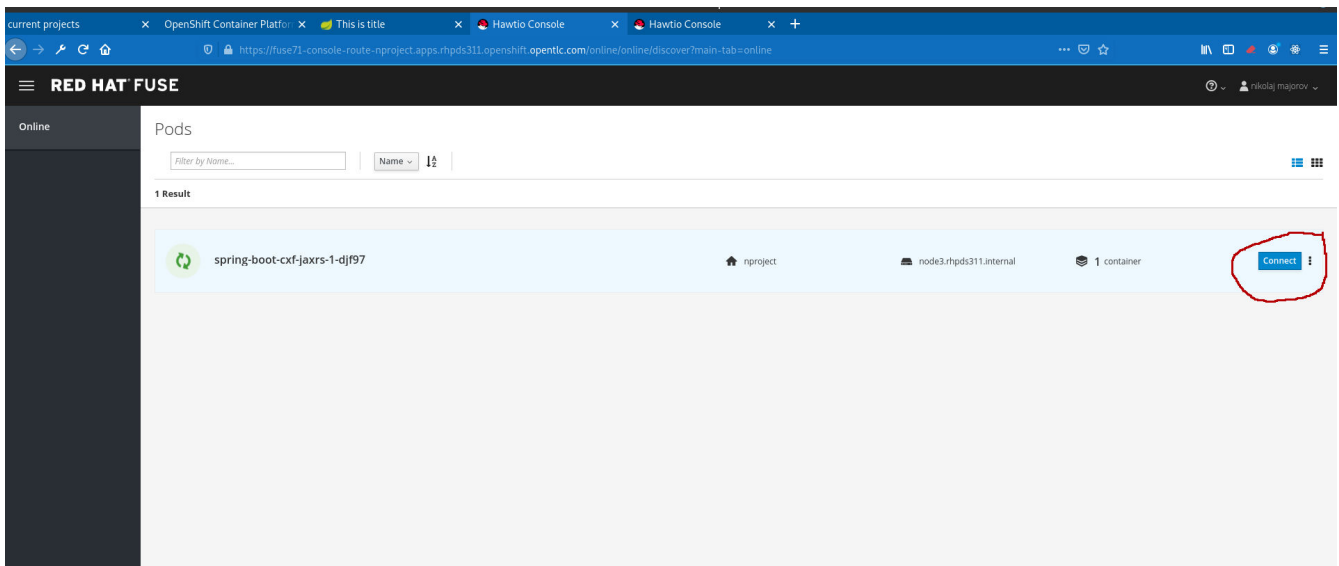
install template if it is not present:

```

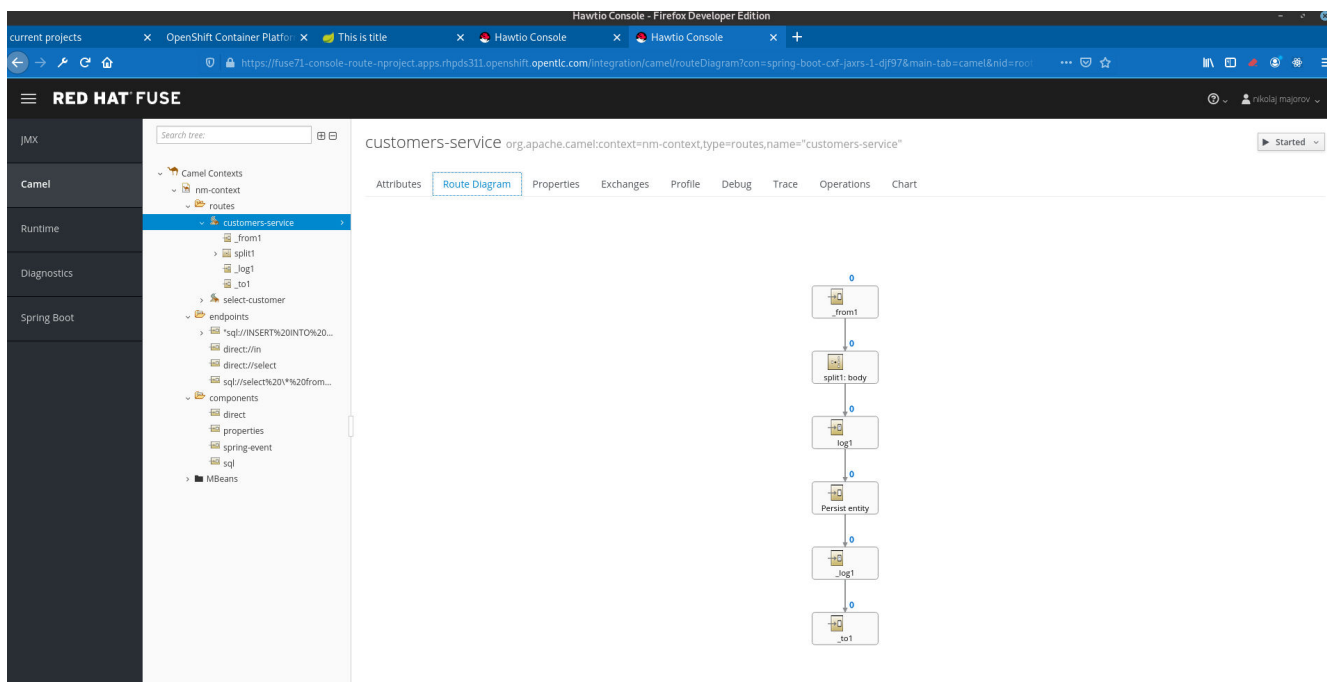
oc create \
  -n openshift \
  -f https://raw.githubusercontent.com/jboss-fuse/application-templates/application-templates-2.1.fuse-000099-redhat-5/fis-console-namespace-template.json

```

login to fuse console with OpenShift credentials first, then select running fuse pod:



by clicking connect button you can connect and see the route of specific pod:



In the fuse console you can evaluate statistic of camel route and exam the jndi tree of the application.

#### 4.3.3. CodeReady Studio

Red Hat recommend [CodeReady studio](#) and [CodeReady Toolchain](#) for local development on laptops.

View on Camel route opened in CodeReady studio:

The screenshot displays the Red Hat CodeReady Studio interface. The top menu bar includes File, Edit, Navigate, Search, Project, Run, Window, and Help. The toolbar contains various icons for file operations, navigation, and execution. The left sidebar shows the Project Explorer with the following structure:

- fuse7-code-first-approach [fuse7-code-first-approach]
  - Camel Contexts
    - src/main/java
    - src/test/kotlin
    - src/main/kotlin
    - src/main/resources
      - META-INF
        - spring
          - camel-context.xml
            - Route customers-service
            - Route select-customer
    - static
      - index.html
    - application.properties
    - schema-hsqldb.sql

The JMX Navigator on the left shows a search bar and a list of connections: Local Processes, Server Connections, and User-Defined Connections.

The main workspace displays the `*camel-context.xml` file. It contains two routes:

- Route customers-service**:
  - Starts with `direct:in`.
  - Followed by a `Split split1` component.
  - Then a `Log log1` component.
  - Then an SQL component: `sql:INSERT INTO EXPENS...`.
  - Then another `Log _log1` component.
  - Ends with a `direct:select` component.
- Route select-customer**:
  - Starts with `direct:select`.
  - Followed by an SQL component: `sql:select * from EXPENS...`.
  - Ends with a `Log _log2` component.

The bottom right pane shows the OpenShift console with a list of services and pods. The selected pod is `spring-boot-cxf-jaxrs-1-djf97 Pod Running`.

spring-boot-cxf-jaxrs-1-djf97 Pod Running

## 5. Modernize legacy application

### 5.1. Enterprise Service Bus monolith

Companies are facing a problem with modern technology adoption and need to integrate old legacy application which they are struggle maintain and that they have to support in the lifetime from 5-10 years. Lot of this legacy applications are using SOAP as a integration protocol , SAML as security protocol and running on monolith ESB system. Usually ESB is a Top-Down approach there following aspect of services are centralized:

- SOA governance / Endpoint definitions
- Routing
- Protocol transformation
- Mapping
- Content Enrichment and Mapping
- Security

lets see how we can address this with Microservice Architecture and Red Hat Integration Bundle.

### 5.2. SOA governance / Endpoint definitions

Legacy application are using a WSDL (Web Services Description Language ) to describe Web services based on an abstract model of what the service offers.

TODO: ... add modern way...

### 5.3. Routing

Different clients are need different data and microservices are fine-grained APIs which means client need to interact with multiple service. Solution for this will be to have API gateway that is the single entry point for all clients.

3scale API Gateway is the RedHat answer for microservices routing.

Useful links: [https://access.redhat.com/documentation/en-us/red\\_hat\\_3scale\\_api\\_management/2.6/html/administering\\_the\\_api\\_gateway/operating-apicast](https://access.redhat.com/documentation/en-us/red_hat_3scale_api_management/2.6/html/administering_the_api_gateway/operating-apicast)

### 5.4. Protocol transformation

Example of SOAP protocol transformation

[https://github.com/nmajorov/fuse\\_eap\\_keycloak](https://github.com/nmajorov/fuse_eap_keycloak)

**WSDL to Rest Migration Tool** helped to migrate from legacy service definitions.

More infos can be found here: <https://github.com/jboss-fuse/wsd2rest>