REMOTE MANAGEMENT AND MONITORING OF DISTRIBUTED OSGI APPLICATIONS

TIM VERBELEN

Senior Researcher
imec – Ghent University

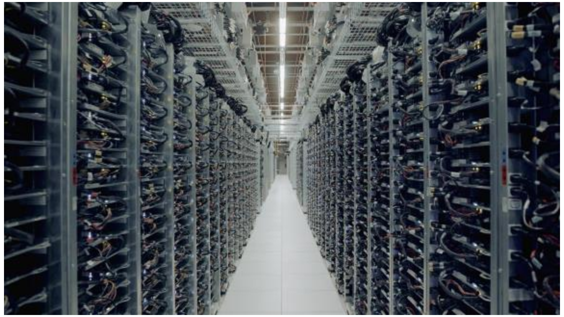
JAN S. RELLERMEYER

Assistant Professor
TU Delft

REMOTE MANAGEMENT AND MONITORING

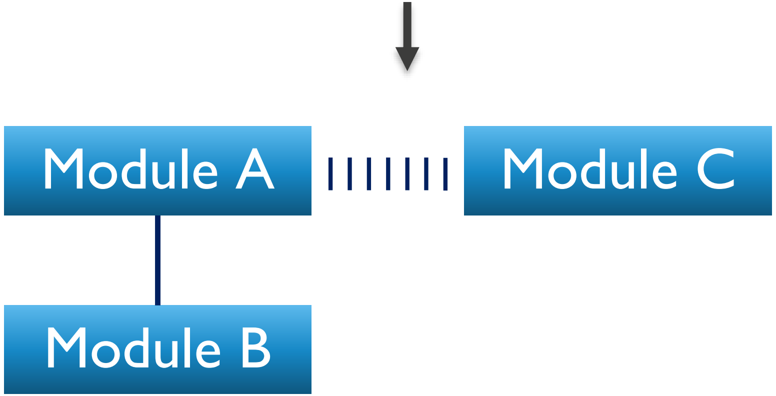
FROM IOT TO CLOUD

- On-site management too expensive
- The device has no screen or input devices
- The device is unreachable
- There are too many devices
- …



OSGI – MODULARITY FOR THE JVM

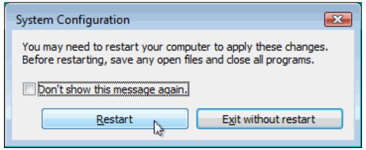
- Modular Software Management
  - Can be managed individually
  - Understand versioning
  - The implementation of modularity for the JVM
    - is the Bundle



- Dynamism
  - Well-defined life-cycle
  - Lose coupling
  - Complex systems are inherently dynamic

OSGI – THE PLATFORM FOR LONG-RUNNING DEVICES

- Java on the Embedded Device
  - Pros:
    - bridges heterogeneity
    - can run the same code in the cloud or on the device
  - Cons:
    - updates and maintenance



System Configuration

You may need to restart your computer to apply these changes.
Before restarting, save any open files and close all programs.

□ Don't show this message again.

- Dynamic Module System for the Java Runtime

- Original domain: Home Gateway
  - Multiple software packages need to co-exist on the same machine
  - Long-running software
  - Periodic maintenance and updates requires
  - Ideally with minimal downtime and no interruption of unrelated software packages

REMOTE MANAGEMENT IS NOT NEW (IN OSGI)

TR-069 CONNECTOR SERVICE SPECIFICATION

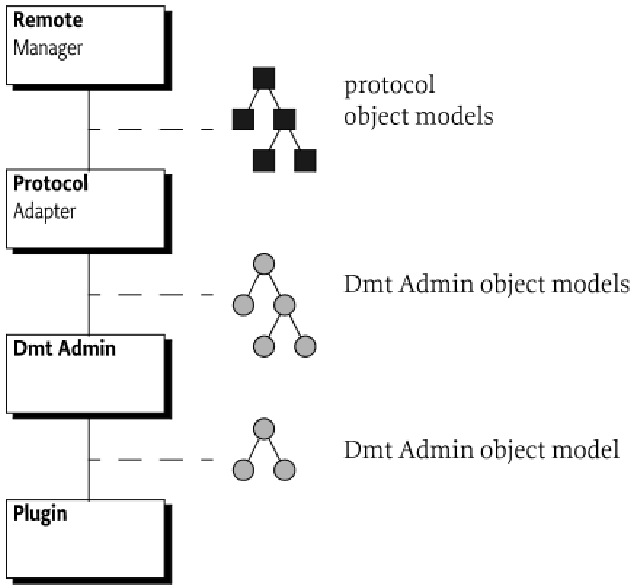
- TR-069 is a remote management protocol specified by the Broadband Forum
- Specifies a bi-directional protocol based on SOAP 1.1 over HTTP

- For managing "Customer-Premise Equipment (CPE)"
- Designed for routers, gateways, set-op boxes,…

REMOTE MANAGEMENT IS NOT NEW (IN OSGI)

DEVICE MANAGEMENT TREE (DMT)

- A logical view of manageable entities implemented by plugins and structured in a tree with named nodes
- One generic API to hide a multitude of remote management protocols
- Used in IoT space (i.e. Bosch SI)



REMOTE MANAGEMENT OVER REST

- RESTful interface
- Exposing the framework and its internal state as resources
  - Bundles
  - Services

- Easy management of OSGi deployments through REST
  - Command line tools
  - Web interfaces
  - Application-level

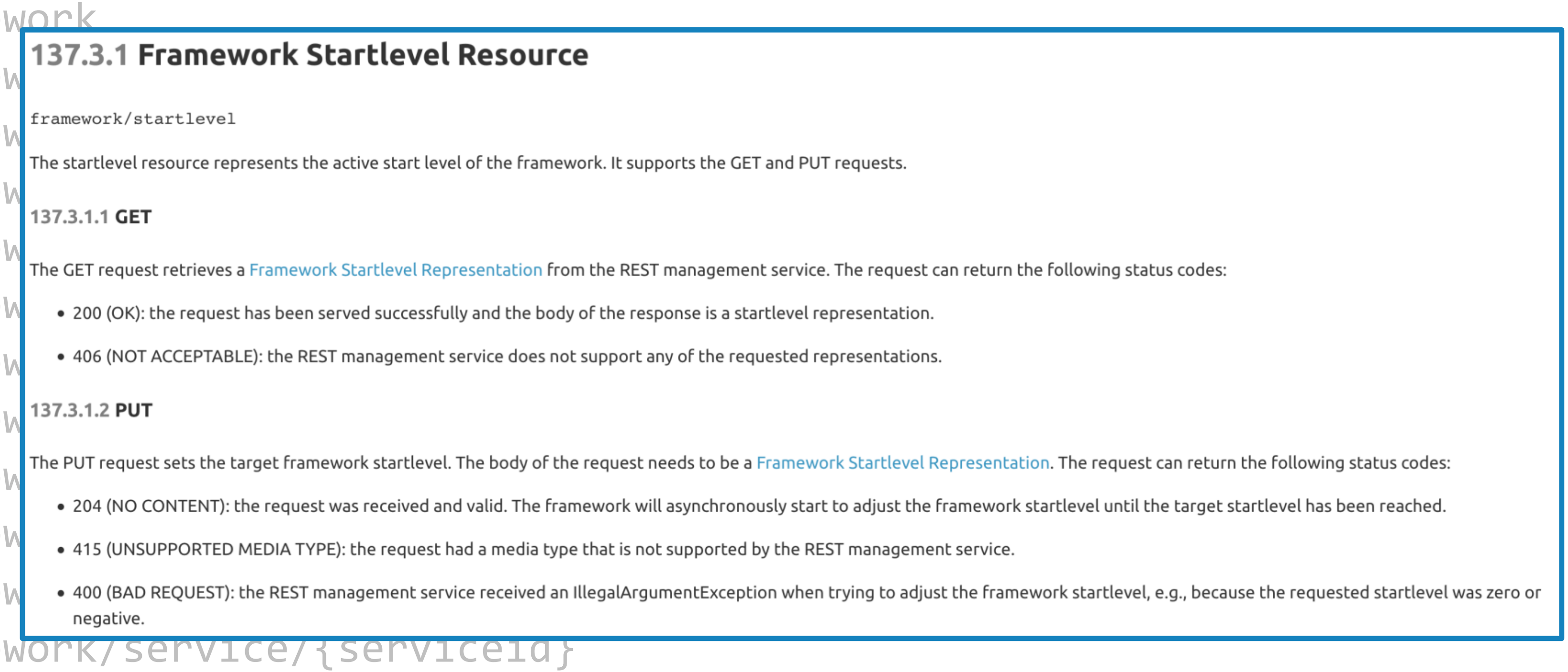
- REST service implementation ships with Concierge 5.1

REST SERVICE ARCHITECTURE

- `framework`
- `framework/state`
- `framework/startlevel`
- `framework/bundles`
- `framework/bundles/representations`
- `framework/bundle/{bundleid}`
- `framework/bundle/{bundleid}/state`
- `framework/bundle/{bundleid}/startlevel`
- `framework/bundle/{bundleid}/header`
- `framework/services`
- `framework/services/representations`
- `framework/service/{serviceid}`

REST SERVICE ARCHITECTURE

- framework
- framew
- framew
- framew
- framew
- framew
- framew
- framew
- framew
- framew
- framew
- framework/service/{serviceid}



137.3.1 Framework Startlevel Resource

`framework/startlevel`

The startlevel resource represents the active start level of the framework. It supports the GET and PUT requests.

137.3.1.1 GET

The GET request retrieves a Framework Startlevel Representation from the REST management service. The request can return the following status codes:

- 200 (OK): the request has been served successfully and the body of the response is a startlevel representation.
- 406 (NOT ACCEPTABLE): the REST management service does not support any of the requested representations.

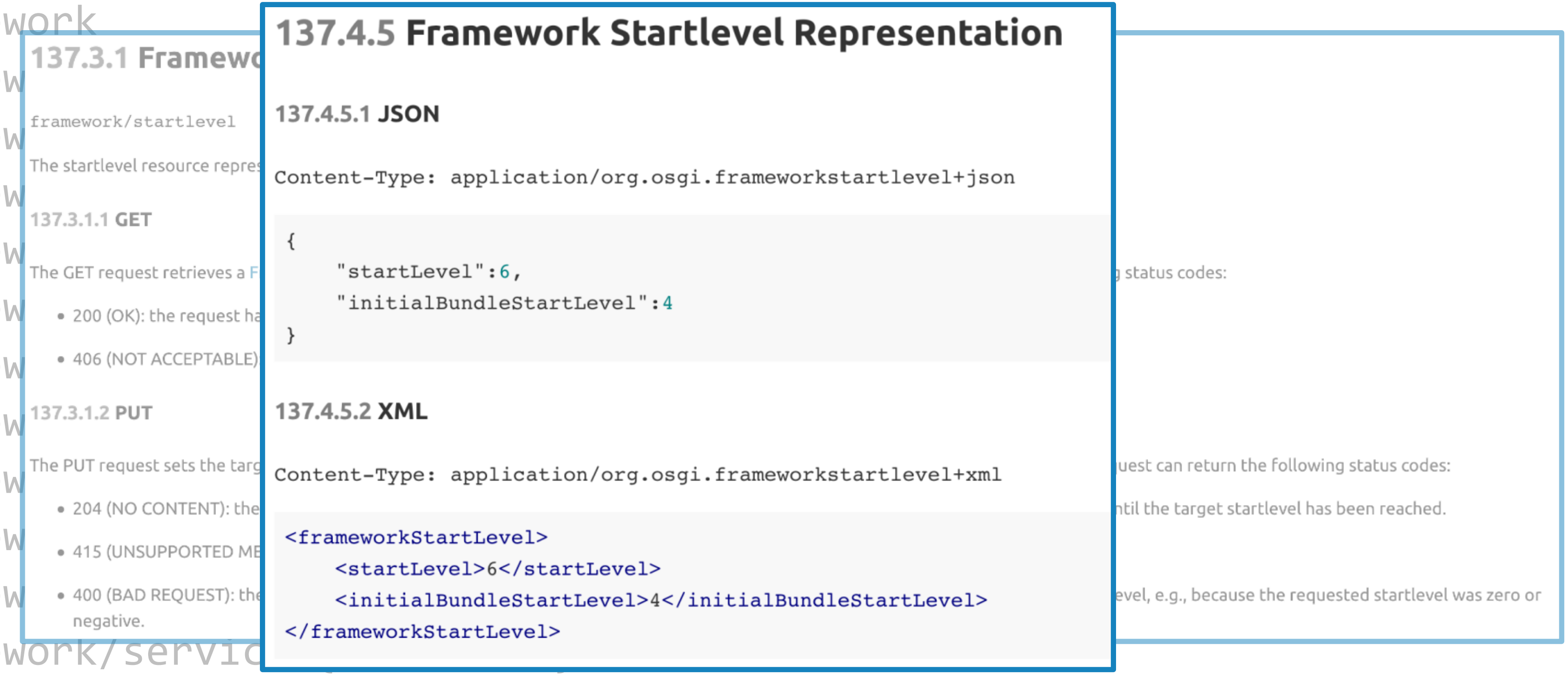
137.3.1.2 PUT

The PUT request sets the target framework startlevel. The body of the request needs to be a Framework Startlevel Representation. The request can return the following status codes:

- 204 (NO CONTENT): the request was received and valid. The framework will asynchronously start to adjust the framework startlevel until the target startlevel has been reached.
- 415 (UNSUPPORTED MEDIA TYPE): the request had a media type that is not supported by the REST management service.
- 400 (BAD REQUEST): the REST management service received an IllegalArgumentException when trying to adjust the framework startlevel, e.g., because the requested startlevel was zero or negative.

REST SERVICE ARCHITECTURE

- framework
- framew
- framew
- framew
- framew
- framew
- framew
- framew
- framew
- framew
- framew
- framework/servic



137.3.1 Framewo

framework/startlevel

The startlevel resource repres

137.3.1.1 GET

The GET request retrieves a F … g status codes:

- 200 (OK): the request ha
- 406 (NOT ACCEPTABLE)

137.3.1.2 PUT

The PUT request sets the targ … uest can return the following status codes:

- 204 (NO CONTENT): the … ntil the target startlevel has been reached.
- 415 (UNSUPPORTED ME
- 400 (BAD REQUEST): the … evel, e.g., because the requested startlevel was zero or negative.

137.4.5 Framework Startlevel Representation

137.4.5.1 JSON

Content-Type: application/org.osgi.frameworkstartlevel+json

```
{
    "startLevel":6,
    "initialBundleStartLevel":4
}
```

137.4.5.2 XML

Content-Type: application/org.osgi.frameworkstartlevel+xml

```
<frameworkStartLevel>
    <startLevel>6</startLevel>
    <initialBundleStartLevel>4</initialBundleStartLevel>
</frameworkStartLevel>
```

REST CLIENT

- RestClientFactory

```
public interface RestClientFactory {
   RestClient createRestClient(URI uri);
}
```

- RestClient

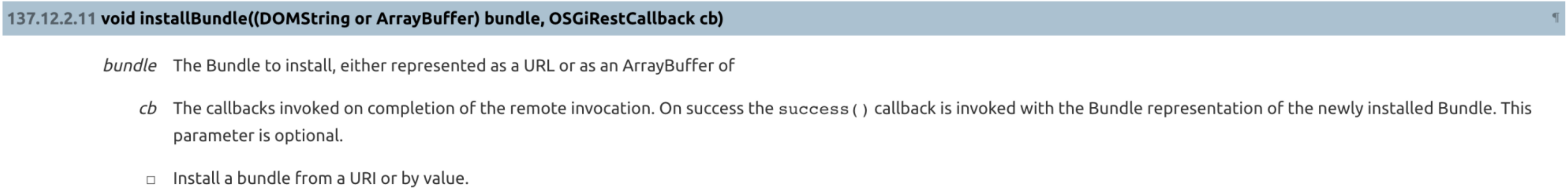
```
public interface RestClient {
   …
   FrameworkStartLevelDTO getFrameworkStartLevel() throws Exception;
   void setFrameworkStartLevel(FrameworkStartLevelDTO startLevel) throws Exception;
   …
}
```

REST CLIENT IN .JS

```
var client = new OSGiRestClient('http://localhost:8080/restendpoint’);
client.installBundle(bundle, {
     success : function(res) {
          // Start the bundle once the install has finished
          client.startBundle(res.id);
     }, failure : function(httpCode, res) {
          // handle failure
     }
});
```

REST CLIENT IN .JS

```
var client = new OSGiRestClient('http://localhost:8080/restendpoint’);
client.installBundle(bundle, {
     success : function(res) {
          // Start the bundle once the install has finished
          client.startBundle(res.id);
     }, failure : function(httpCode, res) {
          // handle failure
     }
});
```



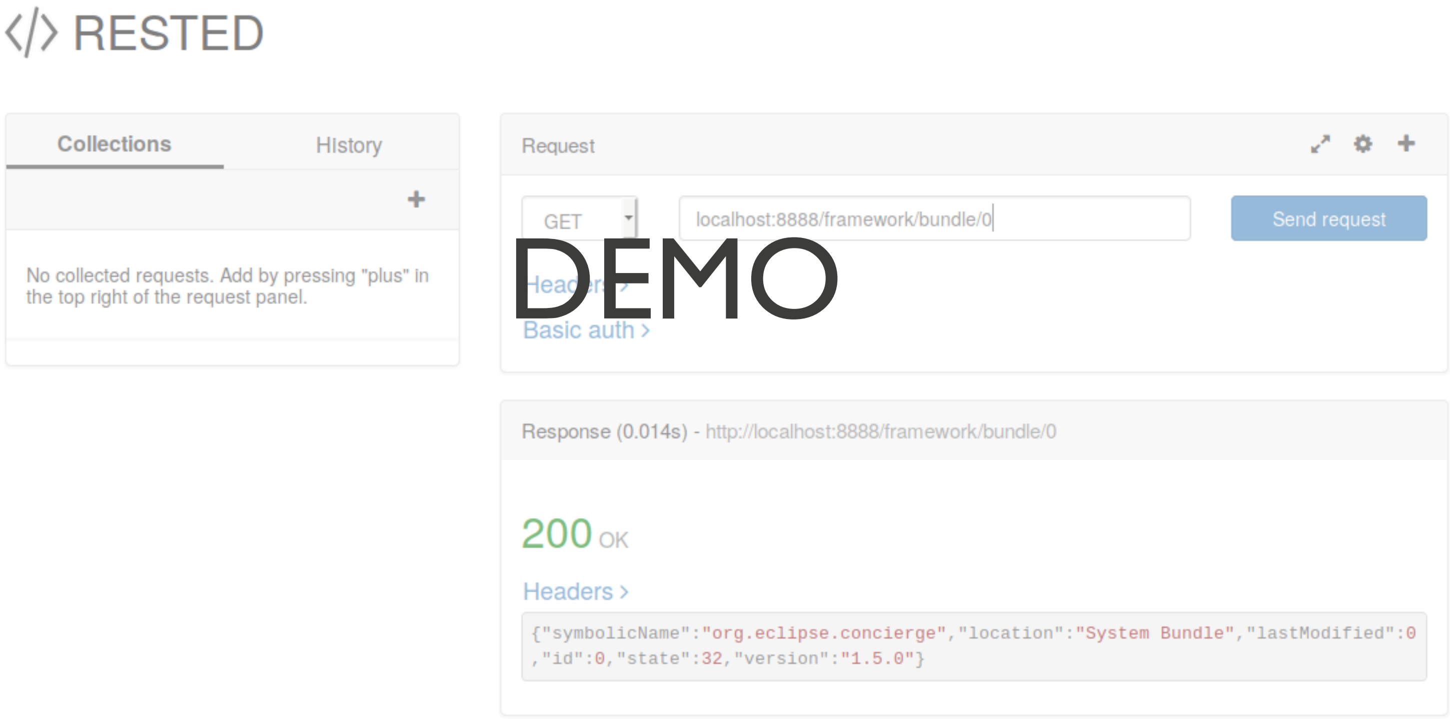
137.12.2.11 **void installBundle((DOMString or ArrayBuffer) bundle, OSGiRestCallback cb)** ¶

bundle The Bundle to install, either represented as a URL or as an ArrayBuffer of

cb The callbacks invoked on completion of the remote invocation. On success the `success()` callback is invoked with the Bundle representation of the newly installed Bundle. This parameter is optional.

□ Install a bundle from a URI or by value.

REMOTE MANAGEMENT OVER REST



No collected requests. Add by pressing "plus" in the top right of the request panel.

localhost:8888/framework/bundle/0

DEMO

Response (0.014s) - http://localhost:8888/framework/bundle/0

200 OK

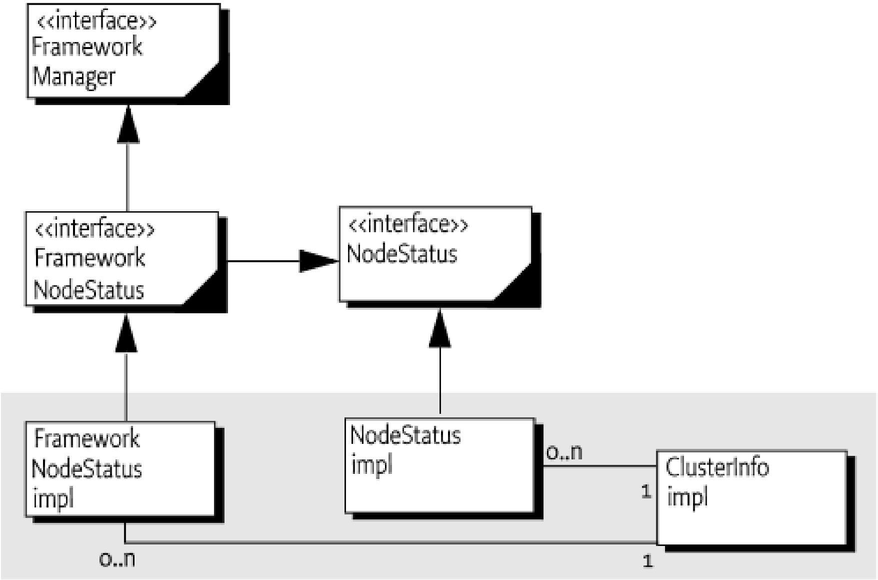
```
{"symbolicName":"org.eclipse.concierge","location":"System Bundle","lastModified":0
,"id":0,"state":32,"version":"1.5.0"}
```

REST SPEC LIMITATIONS

- Requires a REST server on the device to manage
  - Why not use the OSGi service layer? And OSGi Remote Services?
- Limited to a single OSGi framework
  - Why not manage a cluster of OSGi frameworks?
- Limited to OSGi-specific information (i.e. bundles, services, etc)
  - Why no metrics on the underlying (hardware?) platform?

CLUSTER INFORMATION SPECIFICATION

THE NEW KID IN TOWN



FRAMEWORK NODE STATUS

CLUSTER INFORMATION SPECIFICATION

- Presence services indicating a framework / node is there
- Service properties exposing node metadata
  - Unique identifier (and parent)
  - Location information (country, region, zone, …)
  - Endpoints (private and public)
  - Tags (application-specific)

FRAMEWORK NODE STATUS (2)

CLUSTER INFORMATION SPECIFICATION

- `HashMap<String, String> metrics(String... keys)`
  - Provides implementation-specific metrics about the node
  - Typically things that change over time (i.e. resource usage)

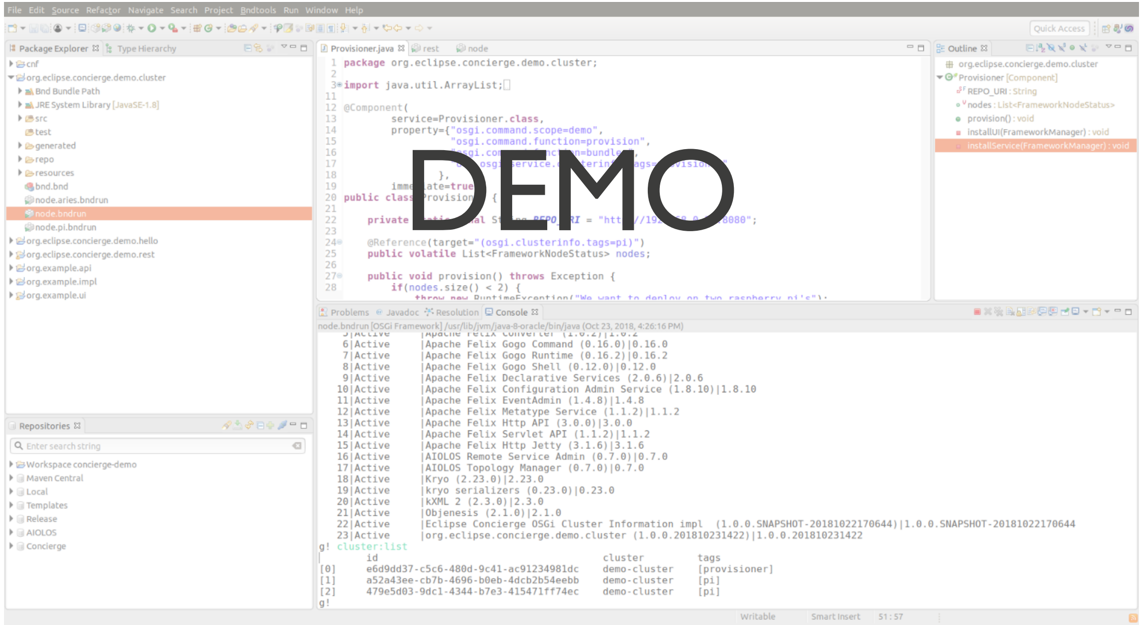
- Integrated with Remote Service Admin specification
  - *NodeStatus exported as remote service
  - Remote services automatically shared within cluster

FRAMEWORK MANAGER

CLUSTER INFORMATION SPECIFICATION

- Managing of an OSGi framework
- Similar API as the REST specification, but now as remote service
- List / inspect bundles and services
- Install / start / update bundles

CLUSTER INFORMATION SPECIFICATION



```java
package org.eclipse.concierge.demo.cluster;

import java.util.ArrayList;

@Component(
        service=Provisioner.class,
        property={"osgi.command.scope=demo",
                  "osgi.command.function=provision",
                  "osgi.command.function=bundles",
                   "org.osgi.service.clusterinfo.tags=provisioner"
                 },
        immediate=true)
public class Provisioner {

    private static final String REPO_URI = "http://192.168.0.[illegible]:8080";

    @Reference(target="(osgi.clusterinfo.tags=pi)")
    public volatile List<FrameworkNodeStatus> nodes;

    public void provision() throws Exception {
        if(nodes.size() < 2) {
            throw new RuntimeException("We want to deploy on two raspberry pi's");
```

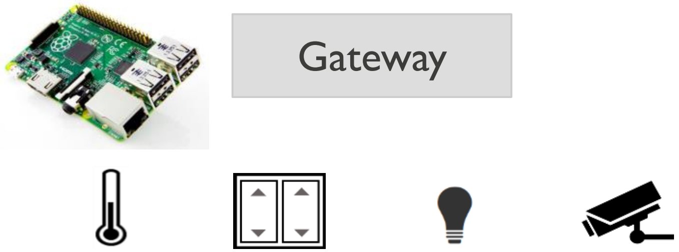
node.bndrun [OSGi Framework] /usr/lib/jvm/java-8-oracle/bin/java (Oct 23, 2018, 4:26:16 PM)

```
    5|Active      |Apache Felix Converter (1.0.2)|1.0.2
    6|Active      |Apache Felix Gogo Command (0.16.0)|0.16.0
    7|Active      |Apache Felix Gogo Runtime (0.16.2)|0.16.2
    8|Active      |Apache Felix Gogo Shell (0.12.0)|0.12.0
    9|Active      |Apache Felix Declarative Services (2.0.6)|2.0.6
   10|Active      |Apache Felix Configuration Admin Service (1.8.10)|1.8.10
   11|Active      |Apache Felix EventAdmin (1.4.8)|1.4.8
   12|Active      |Apache Felix Metatype Service (1.1.2)|1.1.2
   13|Active      |Apache Felix Http API (3.0.0)|3.0.0
   14|Active      |Apache Felix Servlet API (1.1.2)|1.1.2
   15|Active      |Apache Felix Http Jetty (3.1.6)|3.1.6
   16|Active      |AIOLOS Remote Service Admin (0.7.0)|0.7.0
   17|Active      |AIOLOS Topology Manager (0.7.0)|0.7.0
   18|Active      |Kryo (2.23.0)|2.23.0
   19|Active      |kryo serializers (0.23.0)|0.23.0
   20|Active      |kXML 2 (2.3.0)|2.3.0
   21|Active      |Objenesis (2.1.0)|2.1.0
   22|Active      |Eclipse Concierge OSGi Cluster Information impl  (1.0.0.SNAPSHOT-20181022170644)|1.0.0.SNAPSHOT-20181022170644
   23|Active      |org.eclipse.concierge.demo.cluster (1.0.0.201810231422)|1.0.0.201810231422
g! cluster:list
        id                                      cluster         tags
[0]     e6d9dd37-c5c6-480d-9c41-ac91234981dc    demo-cluster    [provisioner]
[1]     a52a43ee-cb7b-4696-b0eb-4dcb2b54eebb    demo-cluster    [pi]
[2]     479e5d03-9dc1-4344-b7e3-415471ff74ec    demo-cluster    [pi]
g!
```

DEMO

ECLIPSE CONCIERGE

- Full OSGi core R5 compatibility
  - R6 almost done
  - R7 to come
- Keep a small footprint to work well on embedded devices
  - 250kiB without debug symbols
  - 330kiB with debug symbols
- Remain “readable”
  - Currently 9 classes
- Remain backwards-compatible
  - Java 5



CONCLUSIONS

- Concierge is an OSGi framework optimized for embedded devices and the Internet of Things
- It hosts the reference implementations for the REST service and the Cluster Information specifications.
- Check them out to ease the remote management of your OSGi deployments.

Project Homepage

http://eclipse.org/concierge

Clone it from here

https://github.com/eclipse/concierge