

# ACE in the Cloud

Karl Pauls & Marcel Offermans

# Karl

- Member Apache Software Foundation
  - PMC: Felix, Sling, Incubator
  - PPMC: Ace, Clerezza, Celix
- Fellow at Luminis
- Co-Author of „OSGi in Action“
- [karl.pauls@luminis.eu](mailto:karl.pauls@luminis.eu)



# Marcel



- **Member Apache Software Foundation**
  - **PMC: Felix, Incubator**
  - **PPMC: ACE, Celix**
- **Fellow at Luminis**
- **[marcel.offermand@luminis.eu](mailto:marcel.offermand@luminis.eu)**

# Luminis

# Luminis



**Luminis Technologies**  
Development and support of innovative software



**Luminis Software Development**  
High-Tech software solutions and services



**Luminis LIVE**  
Designers of digital experiences



**Luminis KIS**  
Knowledge and Information Services



**Luminis CloseSure**  
Safe, certain and close by!

**Carthago ICT**  
Consulting  
ICT projects  
Websolutions

<http://www.luminis.eu/?lang=en>  
<http://luminis-technologies.com/>

# Requirements for exercises

- Eclipse IDE
- Apache Maven
- Java 6
- BndTools
  - at <http://njbartlett.name/bndtools.html>
- E-mail [eclipsecon-2011@luminis.eu](mailto:eclipsecon-2011@luminis.eu) for:
  - access to your EC-2 node with Apache ACE
  - a shared DropBox folder with the exercises

# Agenda

- **Provisioning in OSGi**
- **Hands on: creating an OSGi application**
- **Overview of Apache ACE**
- **Cloud extensions**
- **Hands on: deploying in the cloud**
- **Hands on: building and using ACE locally**
- **Closing Remarks**

# Provisioning in OSGi



# OSGi: core + compendium

**OSGi Service Platform  
Core Specification**  
The OSGi Alliance

Release 4, Version 4.2  
June 2009

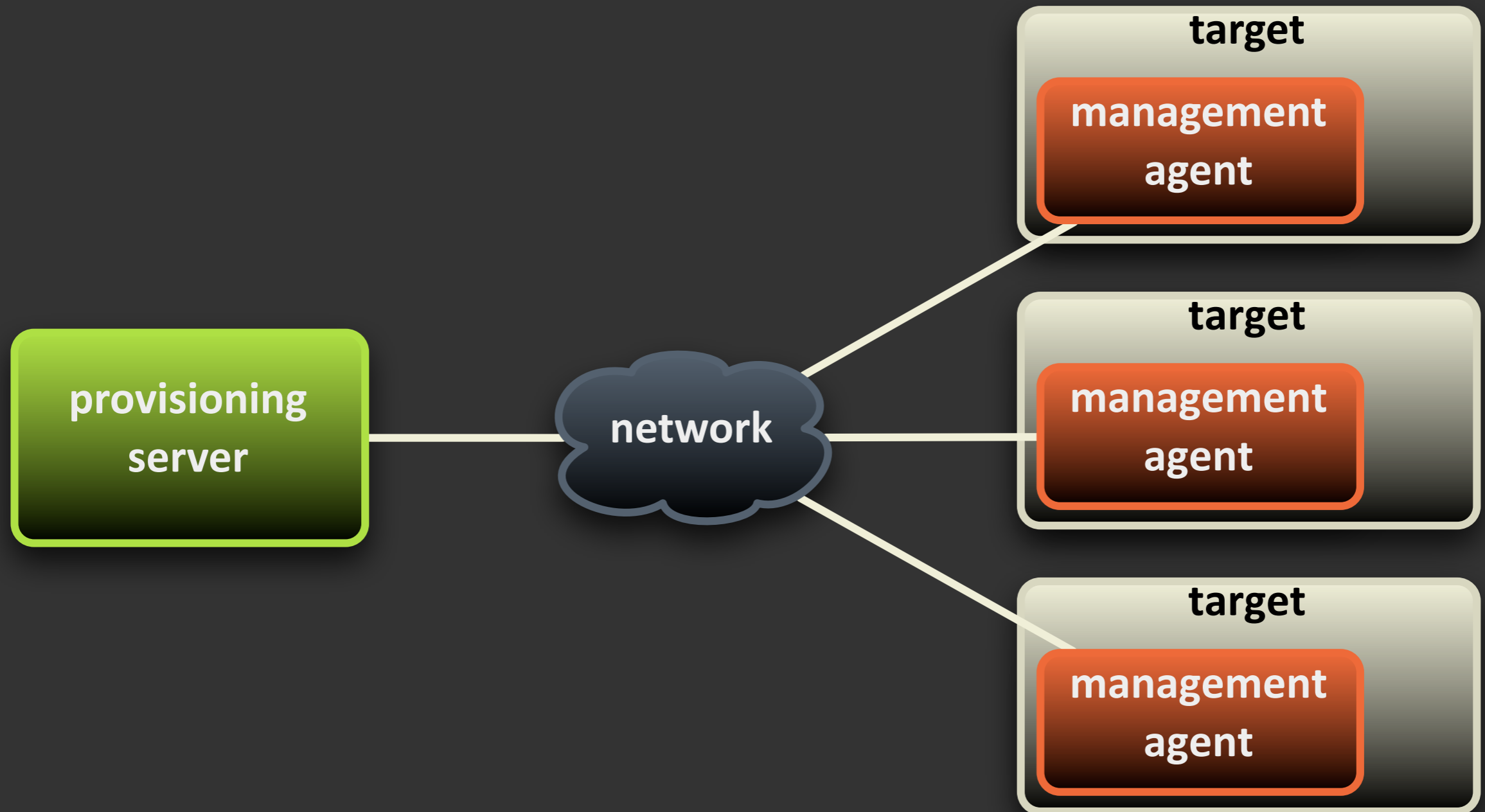


**OSGi Service Platform  
Service Compendium**  
The OSGi Alliance

Release 4, Version 4.2  
August 2009



# Topology



# Management Agent

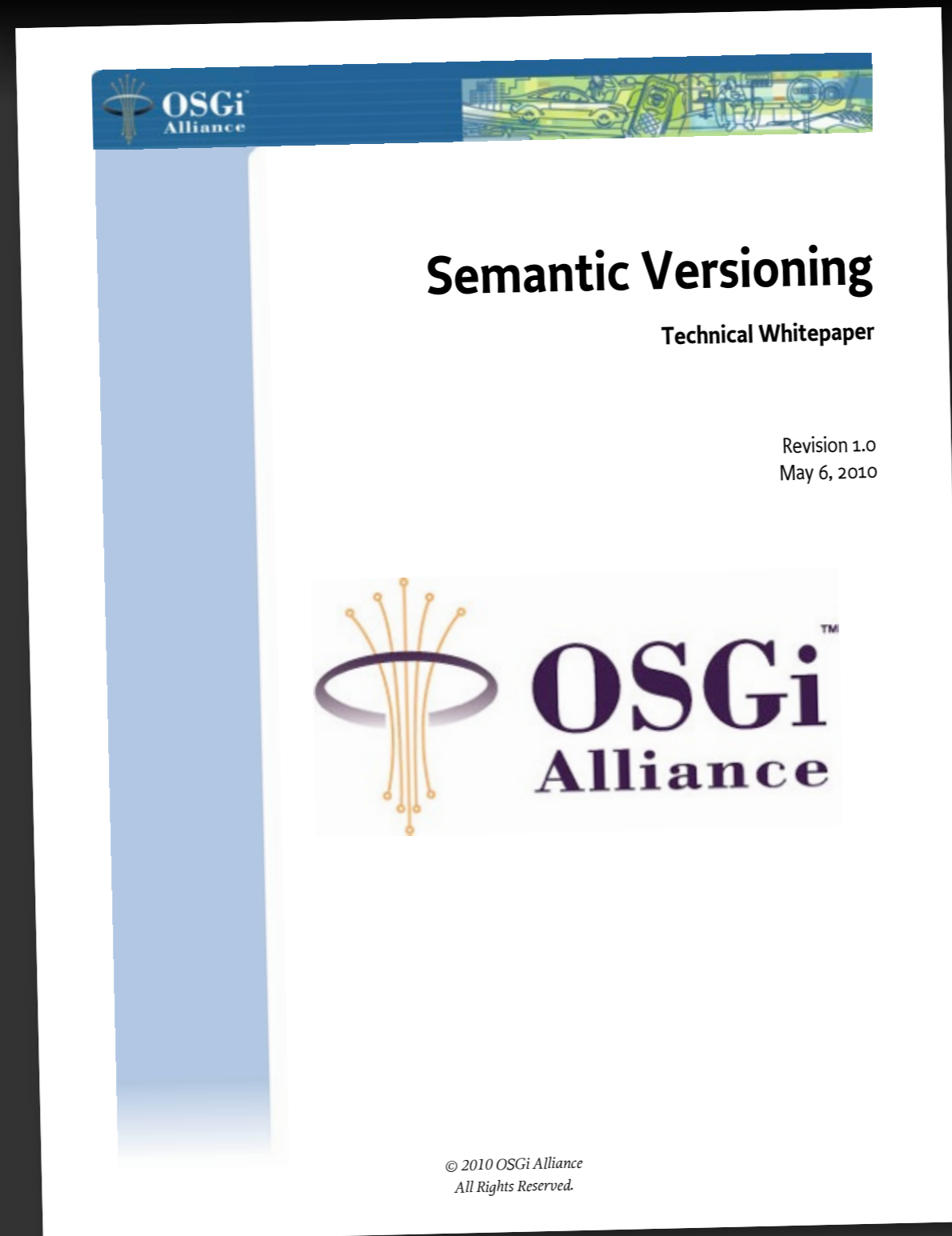
- manages life cycle of bundles  
*BundleContext*
- controls package sharing policies  
*PackageAdmin*
- controls starting/stopping order  
*StartLevel*
- implements a security policy  
*ConditionalPermissionAdmin*

# Management Agent

- manages life cycle of bundles  
*BundleContext*
- controls package sharing policies  
*PackageAdmin*
- controls starting/stopping order  
*StartLevel*
- implements a security policy  
*ConditionalPermissionAdmin*

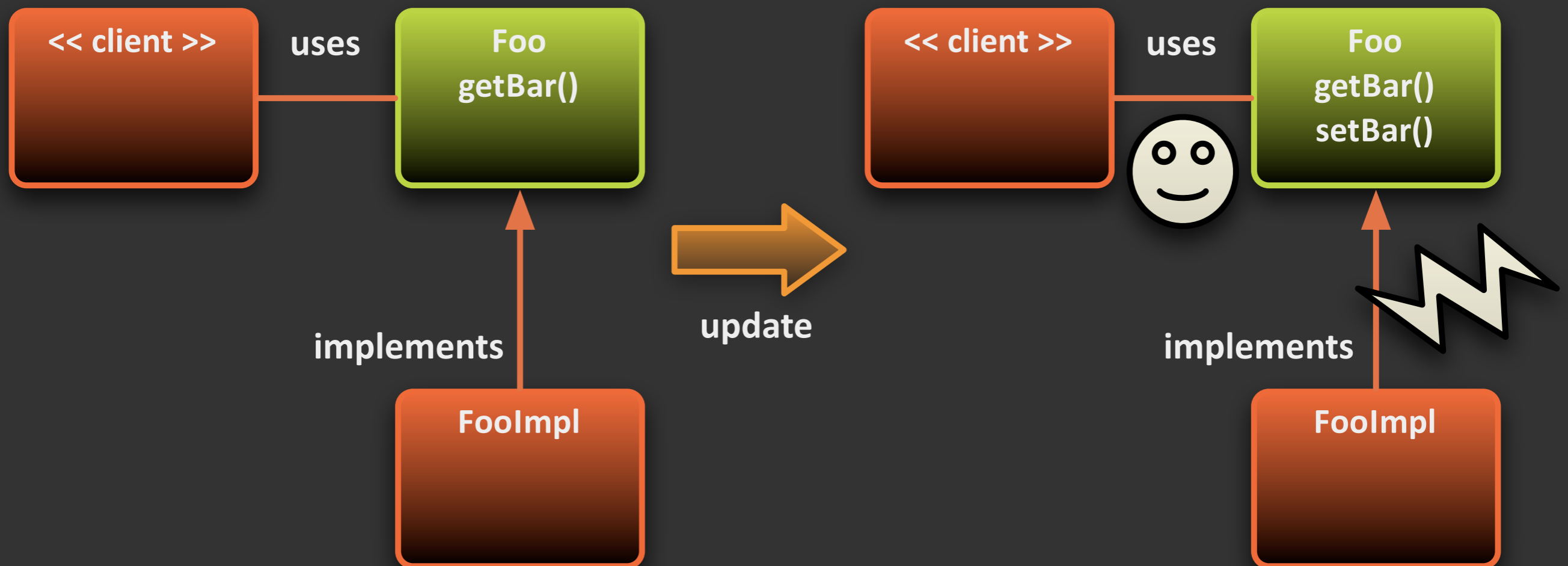


# Tip: Semantic Versioning whitepaper

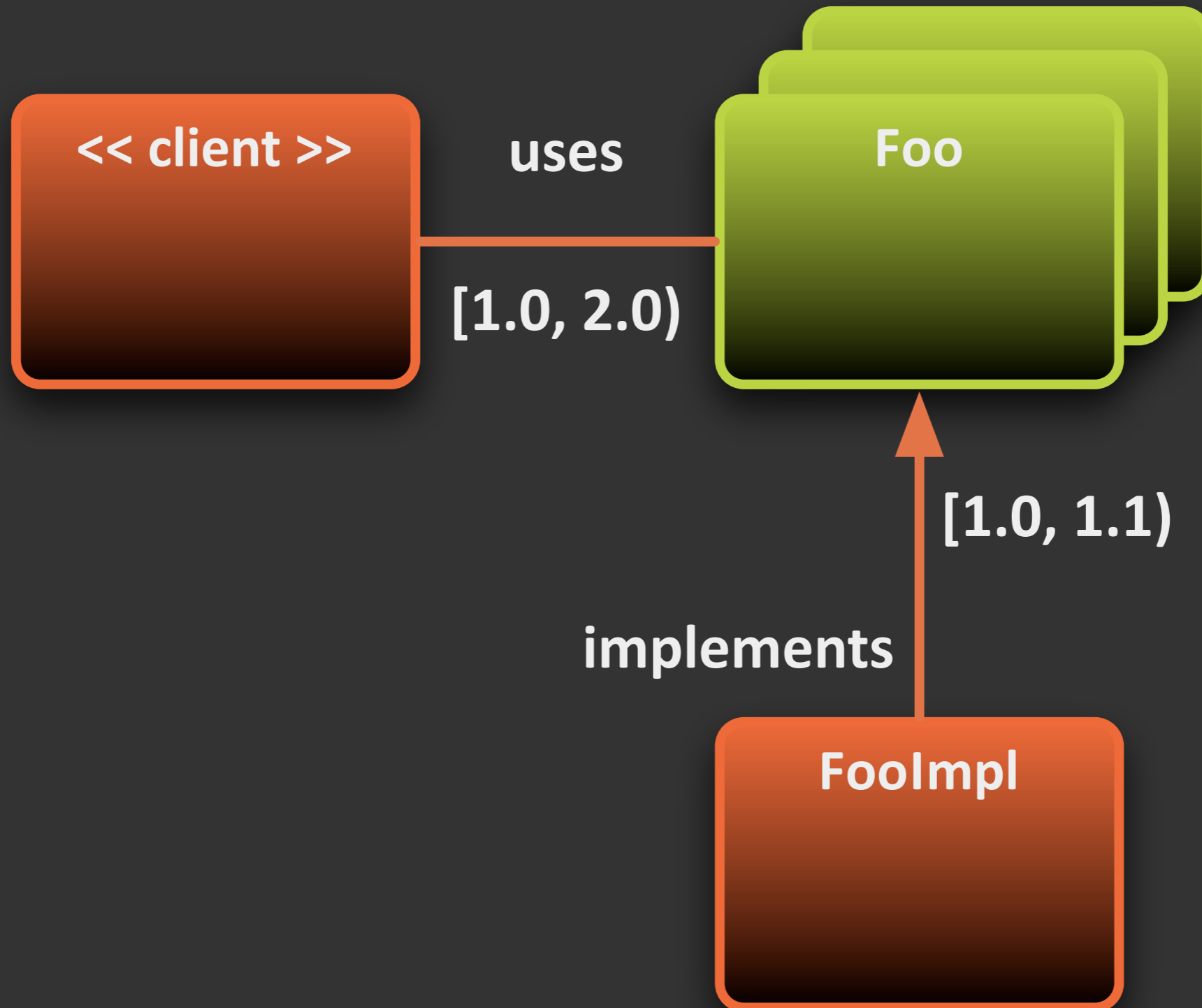


Source: <http://www.osgi.org/wiki/uploads/Links/SemanticVersioning.pdf>

# Downsides and Pitfalls



# Best Practice





**bndtools**



# Bnd

- Tool to create bundles
- .bnd files are “manifests on steroids”
  - Bundle-Version: 1.0.0  
Bundle-Activator: search.solr.Activator  
Private-Package: search.solr,\  
    org.apache.lucene\*, org.apache.solr\*,\  
Import-Package: org.w3c.\*,\  
    javax.xml\*, !junit.\*, !sun.misc,\  
    \*  
Include-Resource: conf
- Details at:  
<http://www.aqute.biz/Code/Bnd>

# BndTools

- Eclipse tools, based on Bnd
- very fast development cycle
- one or more bundles per project
- supports OSGi versioning policies
- Documentation and installation:  
<http://njbartlett.name/bndtools.html>

# Hands On

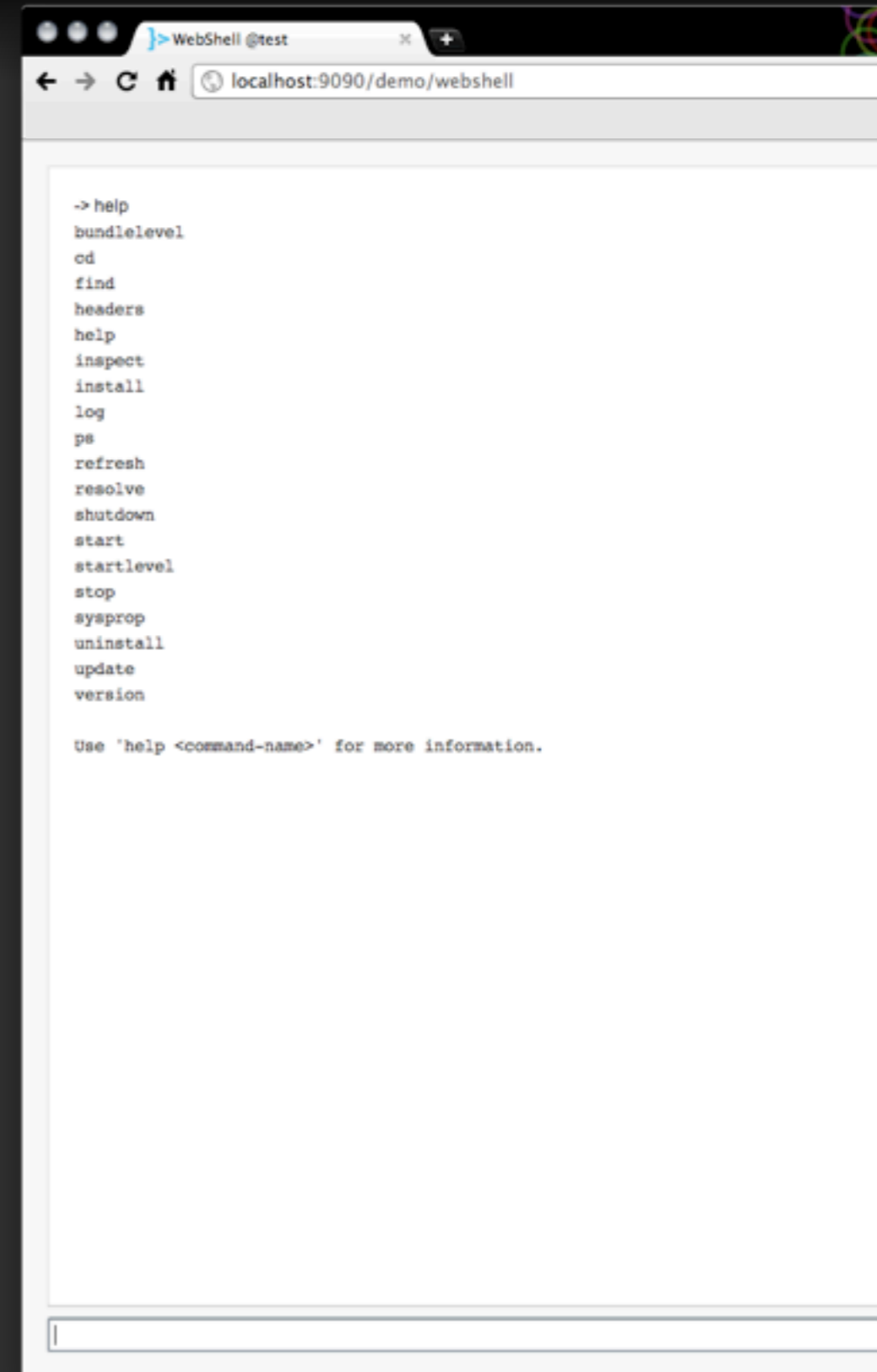
## 1. OSGi application

# Introducing “WebShell”

- **A simple OSGi application that uses the Felix Shell**
- **Web based using the Vaadin UI toolkit**
- **Configurable through Configuration Admin**

# Introducing “WebShell”

- A simple OSGi application that uses the Felix Shell
- Web based using the Vaadin UI toolkit
- Configurable through Configuration Admin



The screenshot shows a web browser window with the title "WebShell @test" and the address bar "localhost:9090/demo/webshell". The main content area displays a list of commands available in the shell:

```
-> help
bundlelevel
cd
find
headers
help
inspect
install
log
ps
refresh
resolve
shutdown
start
startlevel
stop
sysprop
uninstall
update
version
```

Below the list, there is a note: "Use 'help <command-name>' for more information."

# Exercise

- Create a new Eclipse workspace
- Import projects from the projects.zip located on the dropbox folder or memory stick
- Go to webshell/bnd.bnd, in the context menu do “Run As/OSGi Run”
- Point your browser at <http://localhost:9090/demo/webshell>



**Apache**

**ACE**

# Provisioning Solutions

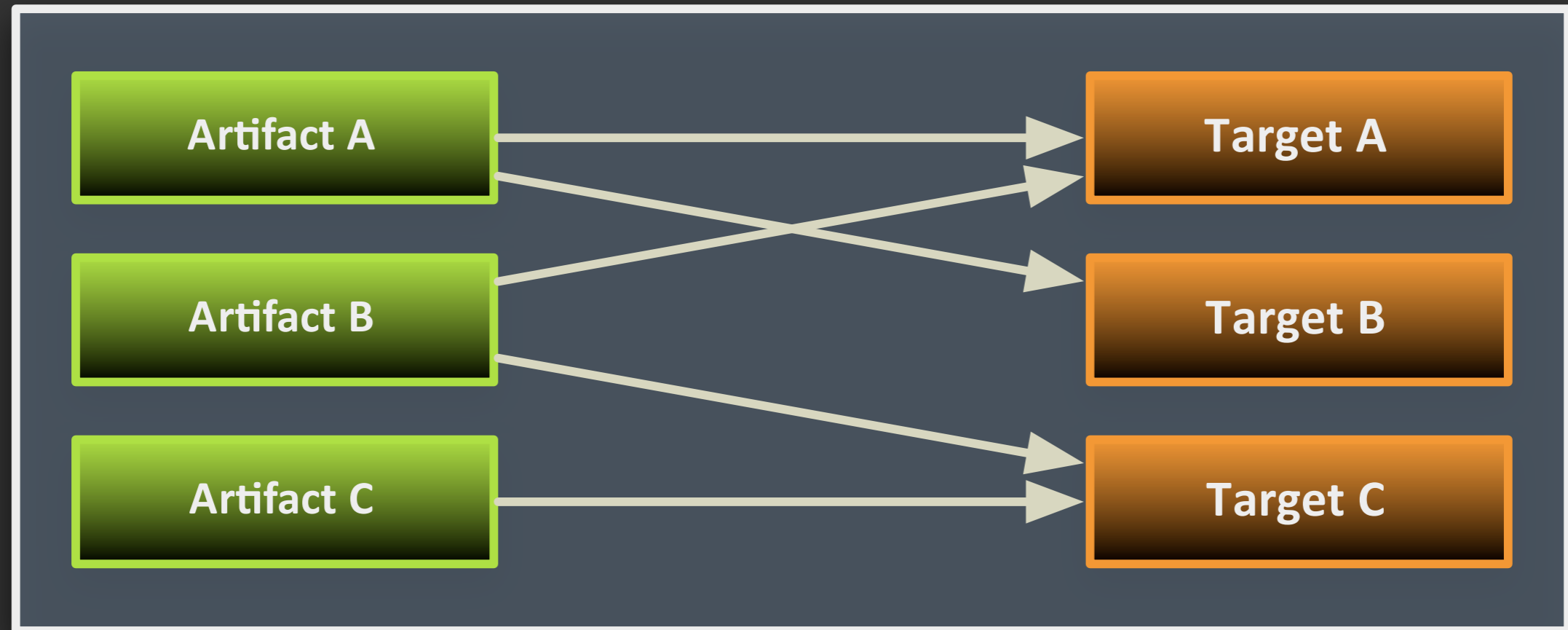
- **Apache Felix File Install**
- **Apache Karaf**
- **Equinox p2**
- **OSGi Bundle Repository Client**
- **Pax Runner**
- **Apache ACE**



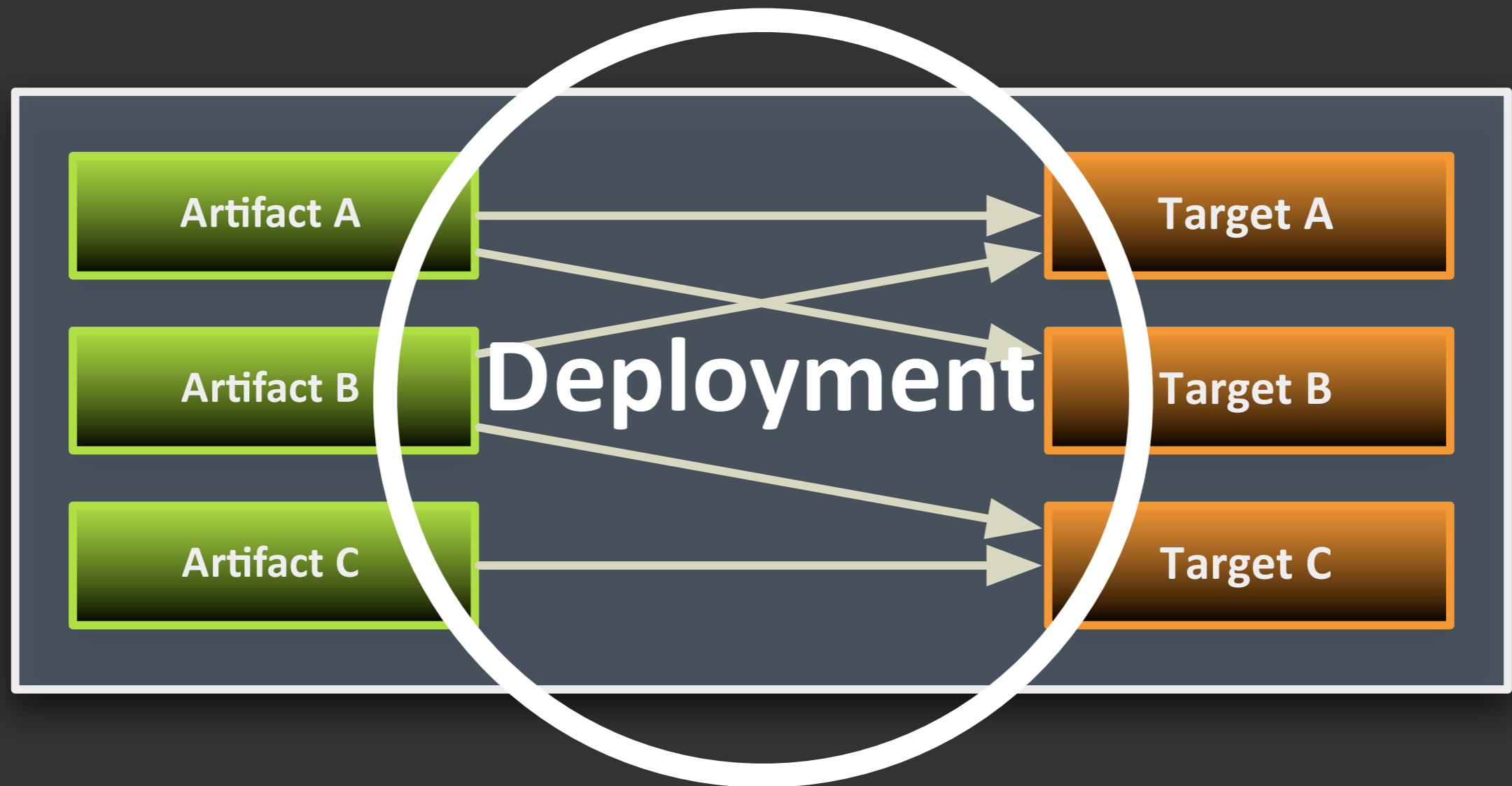
# Apache ACE

- Started in incubator on April 24th 2009
- Software distribution framework based on OSGi
- 12 committers
- working codebase
- <http://incubator.apache.org/ace/>

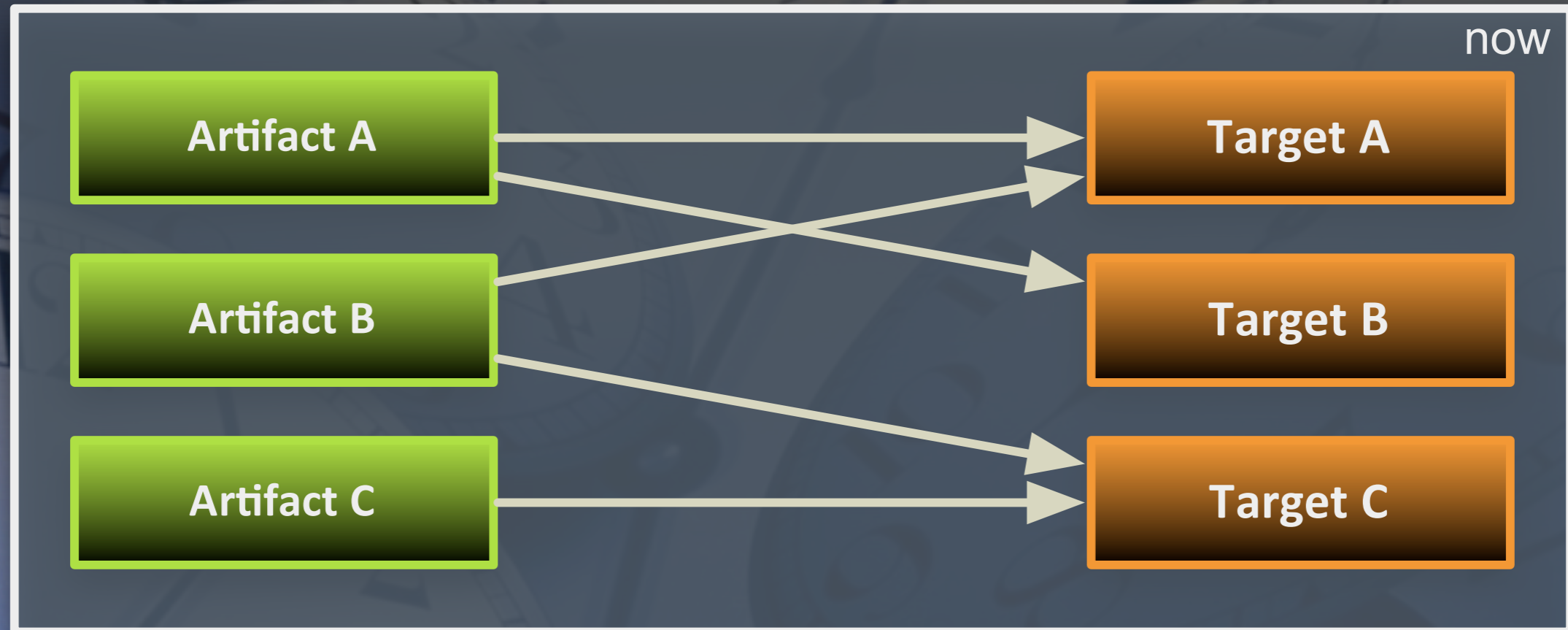
# Deployment



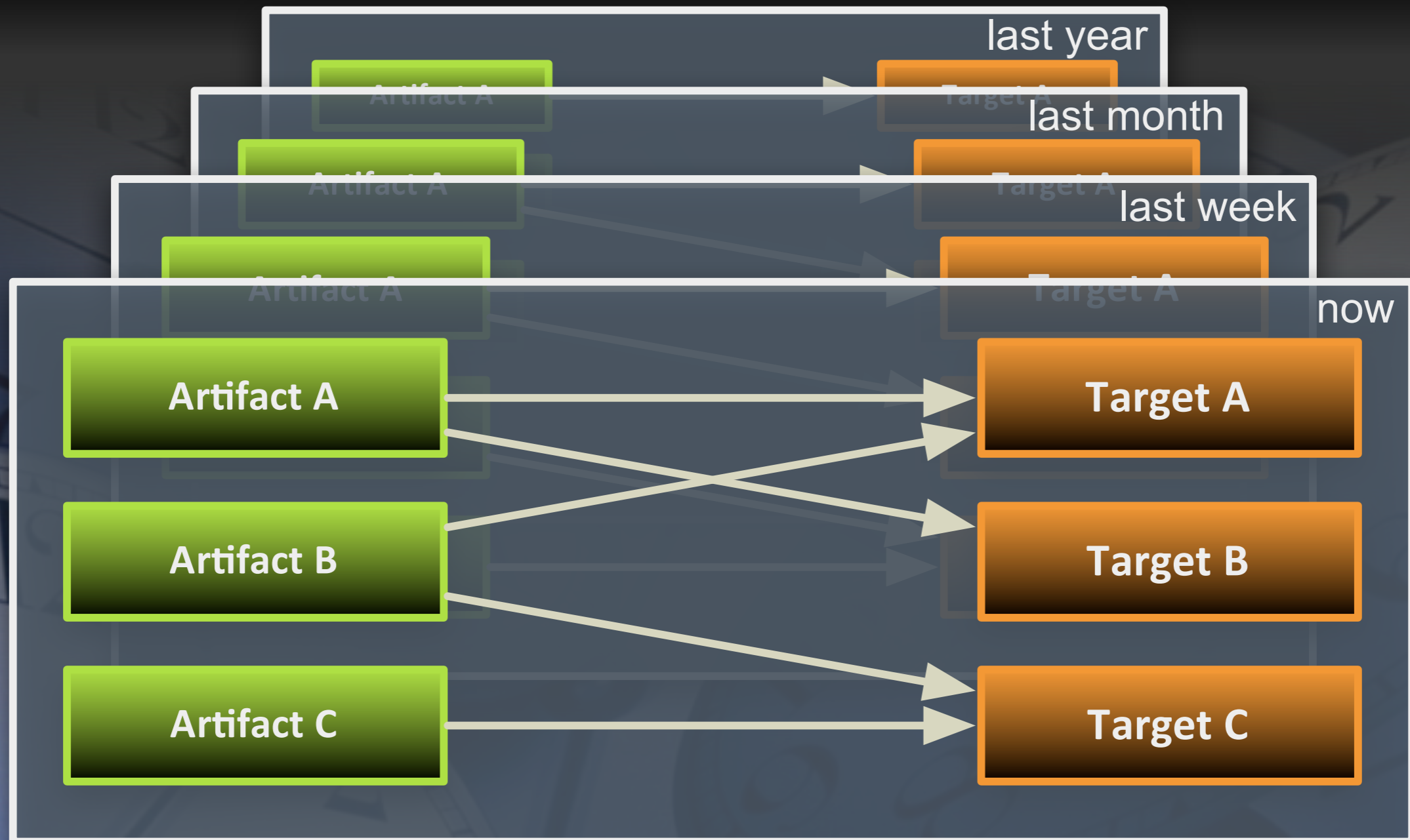
# Deployment



# Keeping the history



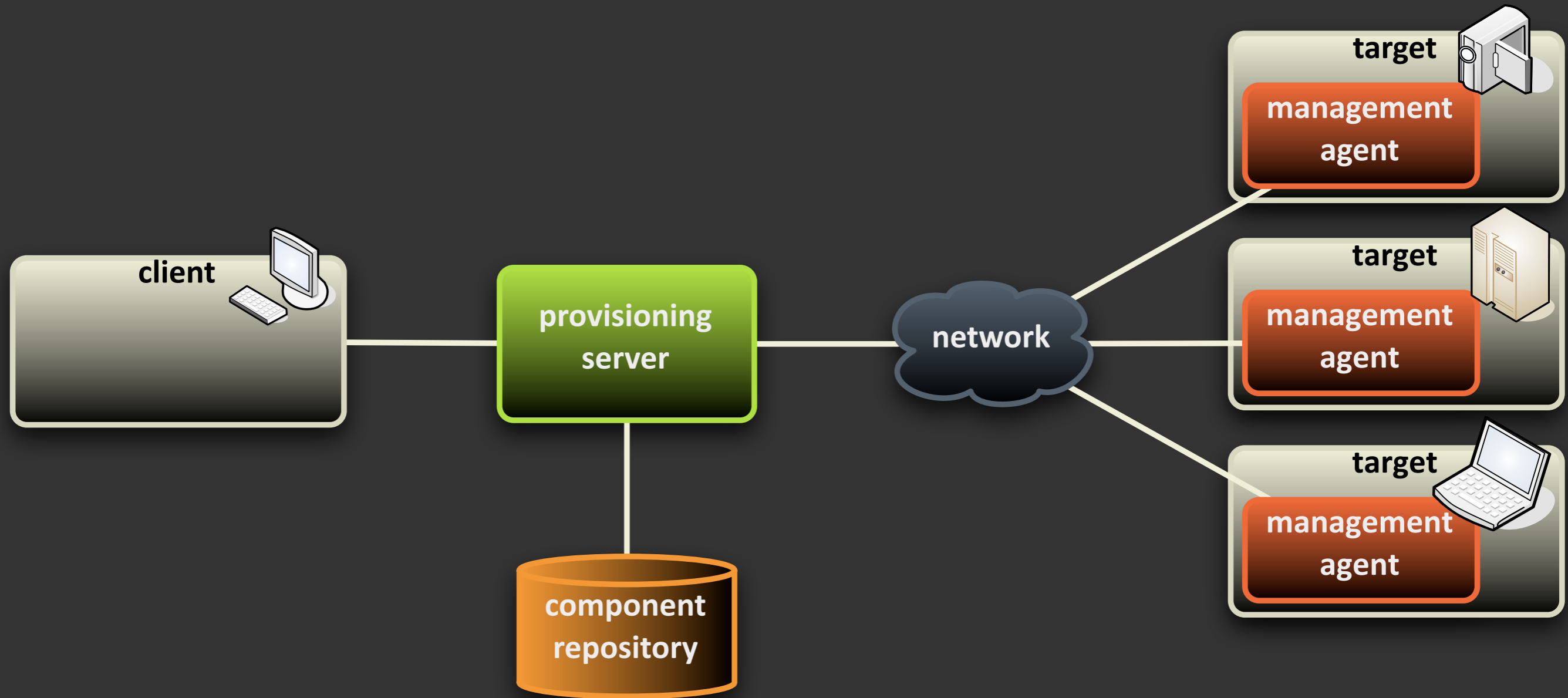
# Keeping the history



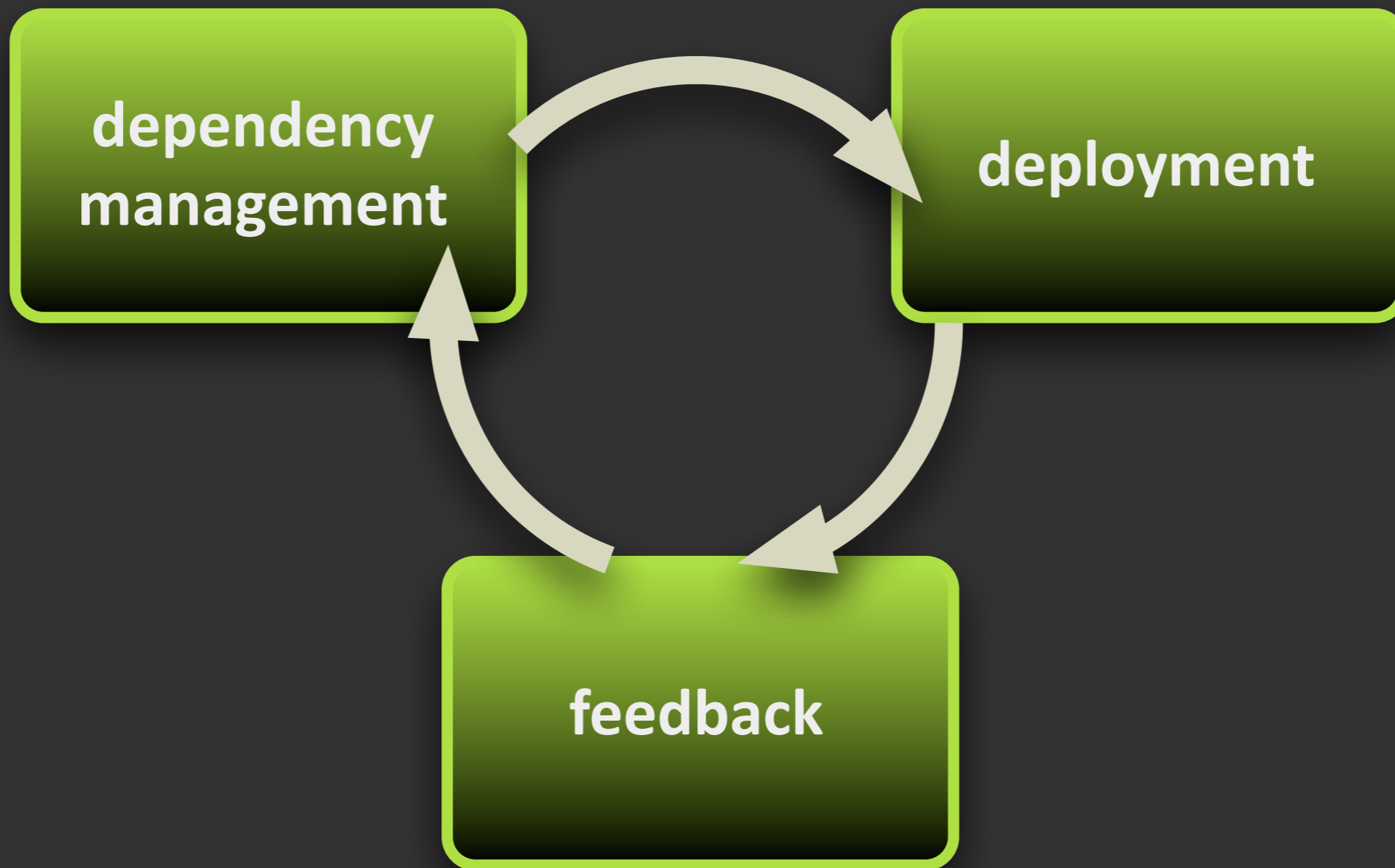
# Why?

- Automate deployment
- Insight into who uses what
- History of each system
- Consistent development, testing, production
- Basis for several possible extensions

# Topology



# High level overview





# High level overview

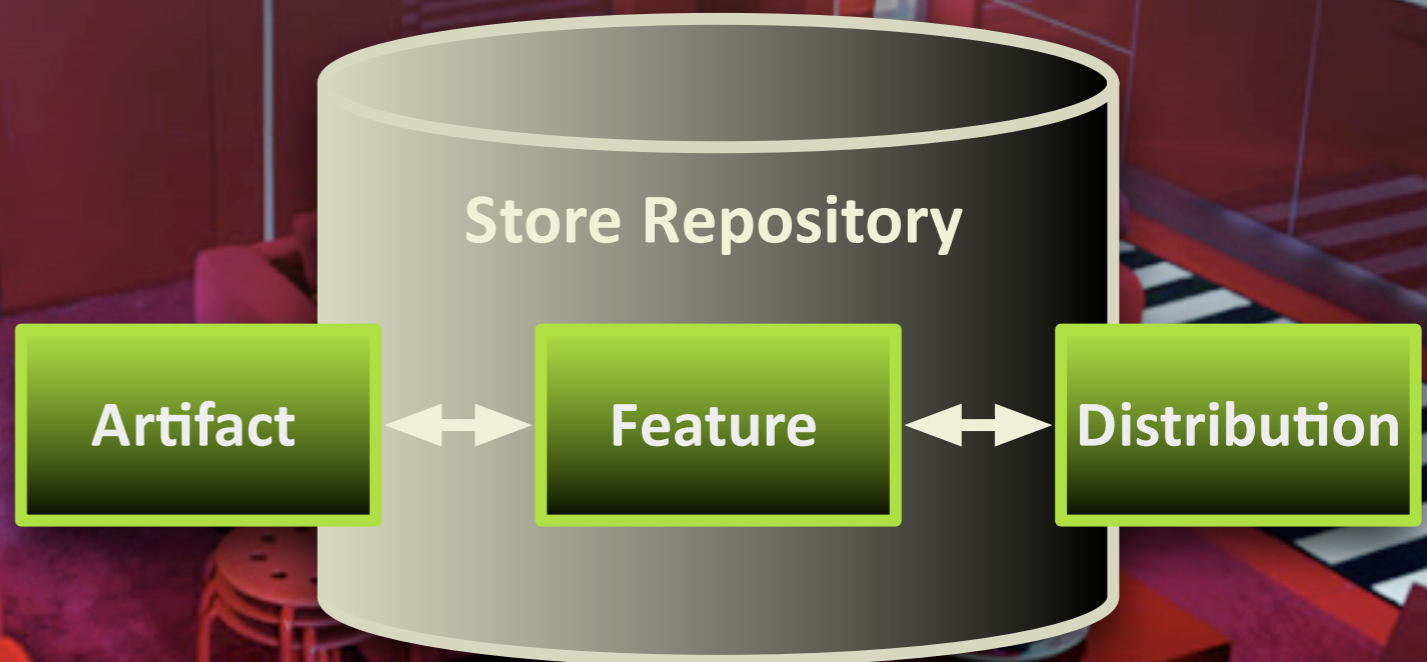
**dependency  
management**

# Dependency Management

- **Organizing artifacts**
- **Mapping them to targets**

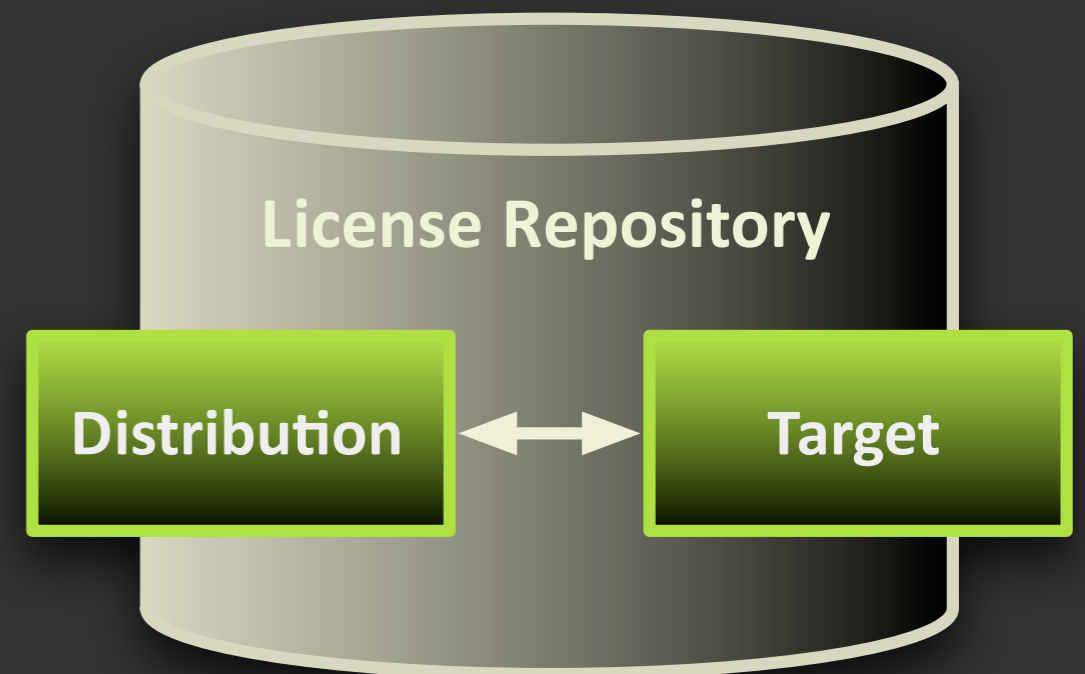
# Organizing artifacts

- group artifacts: makes them manageable
- two levels: feature and distribution
- Analogy: IKEA catalog
- data is kept in “store repository”



# Mapping them onto targets

- mapping distributions to targets
- sometimes done by an external system
- data kept in “license repository”



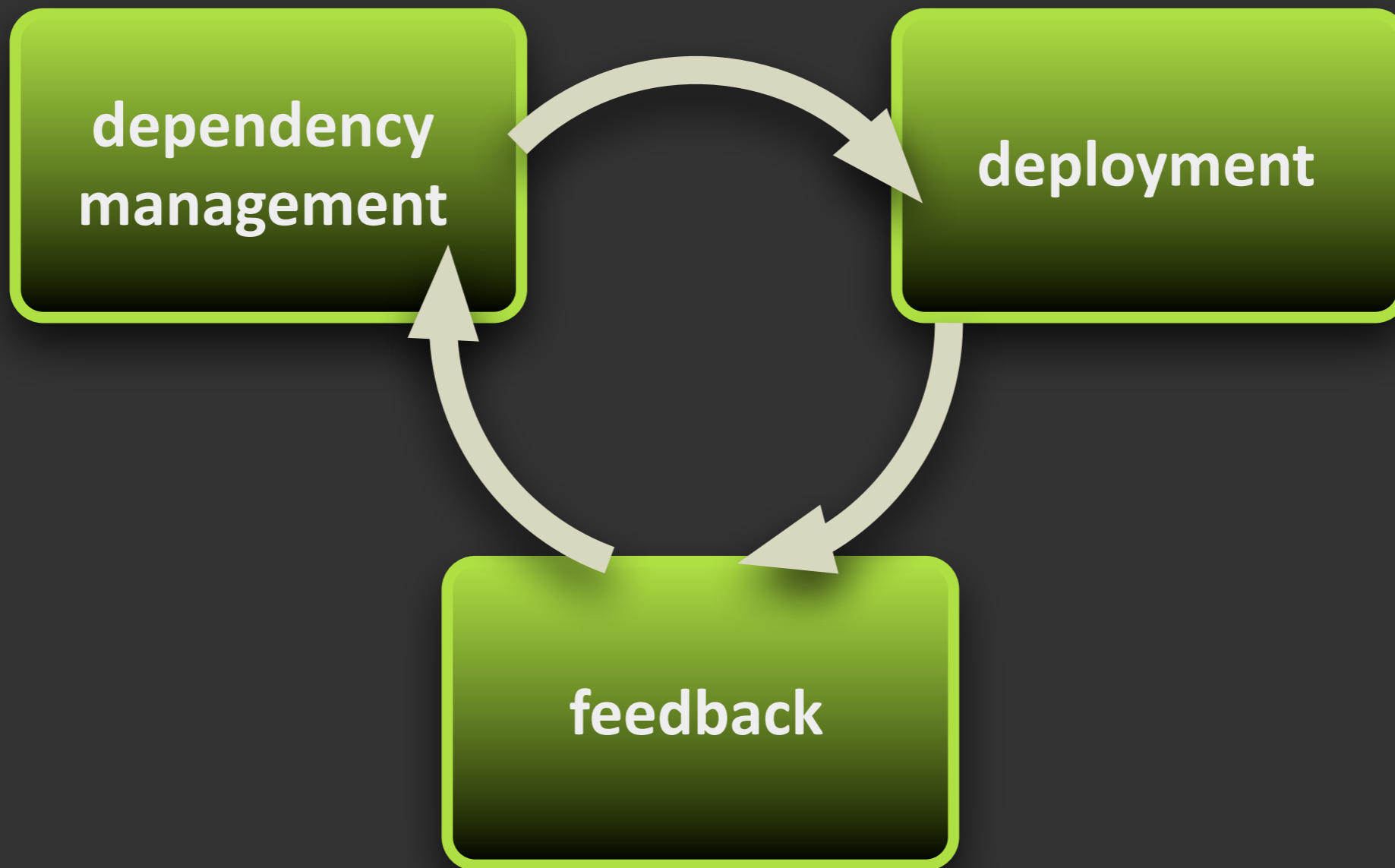
# User Interface

- retrieve, modify and store
- interact with OBR

The screenshot shows the Apache ACE web interface in a browser window. The address bar indicates the URL is localhost:8080/ace. The interface includes several control buttons at the top: Retrieve, Store, Revert, Add artifact..., Add Feature..., Add Distribution..., and Add target... The Dynamic Links checkbox is checked. Below these are four main panels:

- Artifacts:** A list of artifacts with a table header: NAME. The list includes: Apache Felix Gogo Command-0.7.0.SNAPSHOT, Apache Felix Gogo Runtime-0.7.0.SNAPSHOT, Apache Felix Gogo Shell-0.7.0.SNAPSHOT, Apache Felix Service-Based Host-1.0.0, and Apache Felix Triangle Service-1.0.0.
- Features:** A table with headers: NAME, DESCRIPTIO, and ACTIONS. It contains two entries: Gogo Shell (description: the new O;) and Draw App (description: example fr). Each entry has minus and plus action buttons.
- Distributions:** A table with headers: NAME, DESCRIPTION, and ACTIONS. It contains one entry: Default (description: demo distro) with minus and plus action buttons.
- Targets:** A table with headers: NAME and DESCRIPTION. It contains one entry: configuredGatewayID.

# High level overview



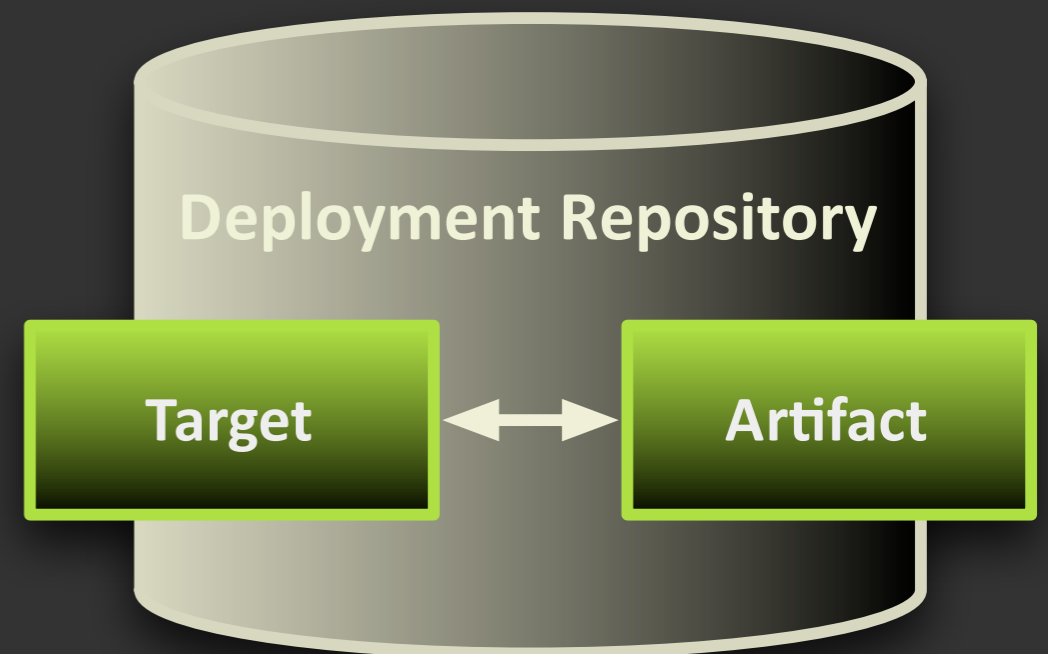
# High level overview

**deployment**



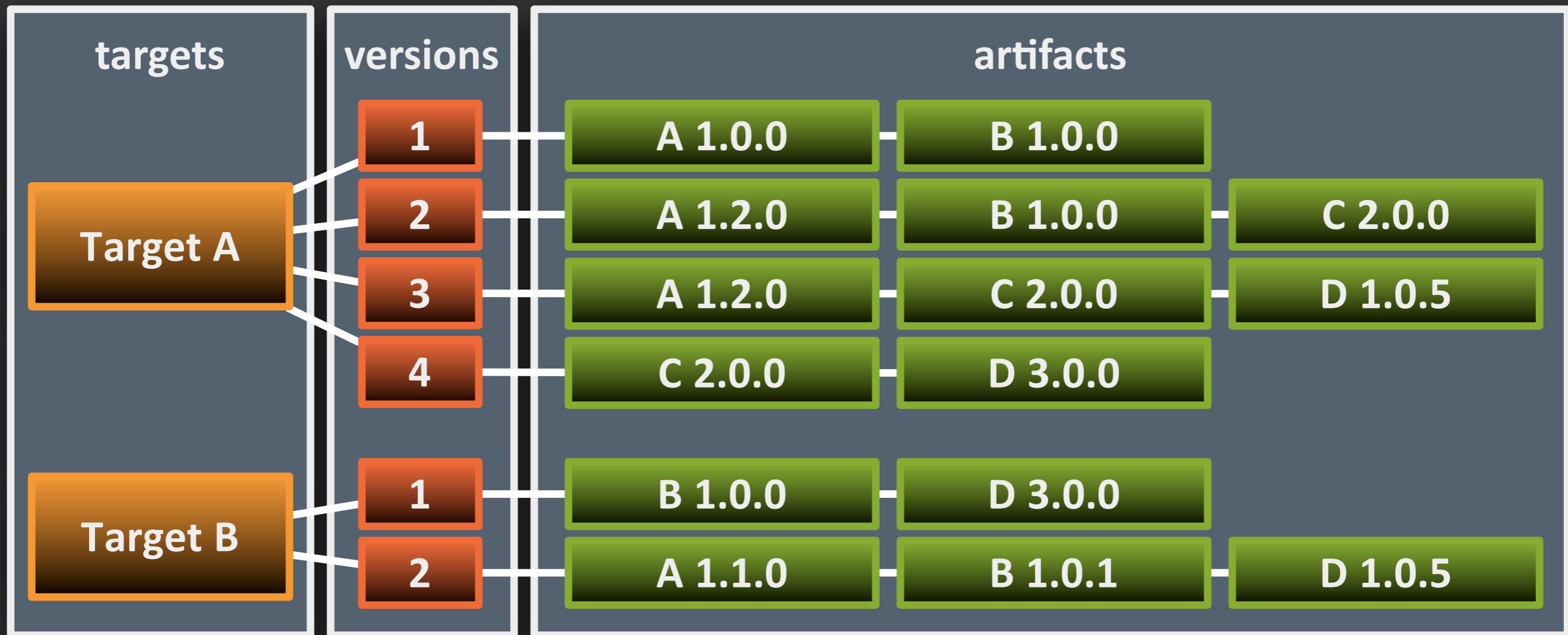
# Deployment

- deployment repository
- management agent





# Deployment Repository



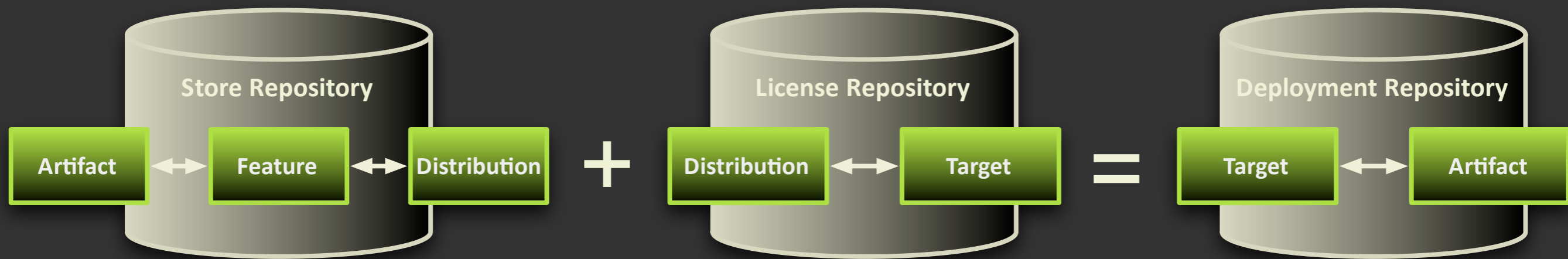
# Management Agent



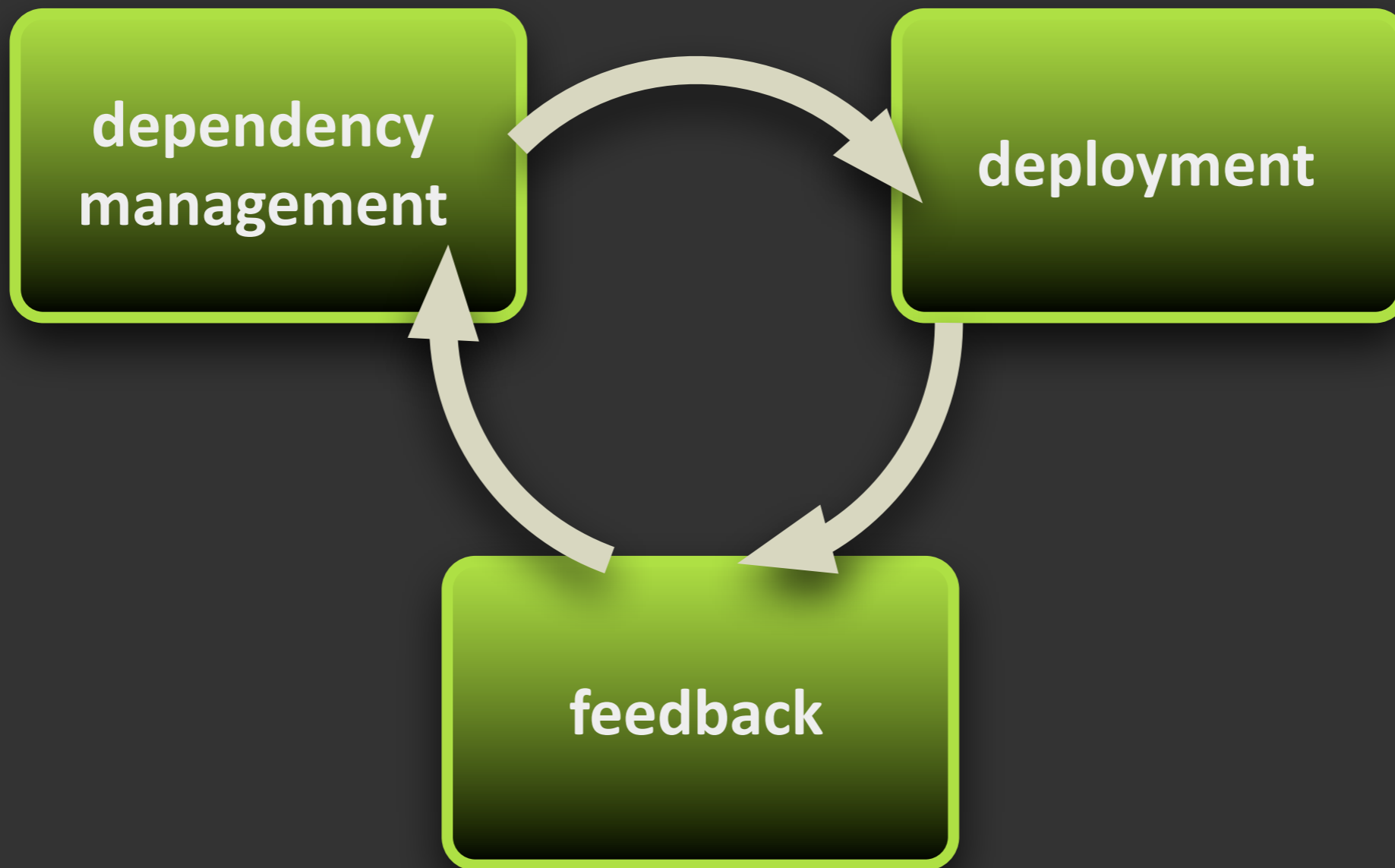
# Deployment Admin

- deployment packages
- versioned set of artifacts
- transactional install/update
- fix packages provide deltas
- signing makes them secure
- extensible through resource processors
- AutoConfig defines configuration admin data

# From dependency to deployment



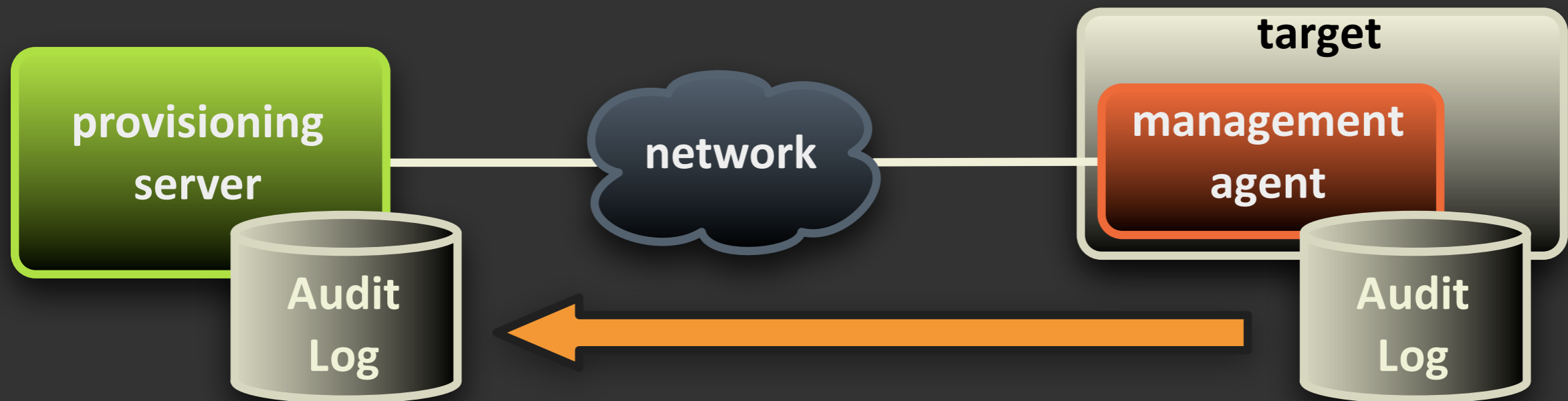
# High level overview



# High level overview

**feedback**

# Feedback



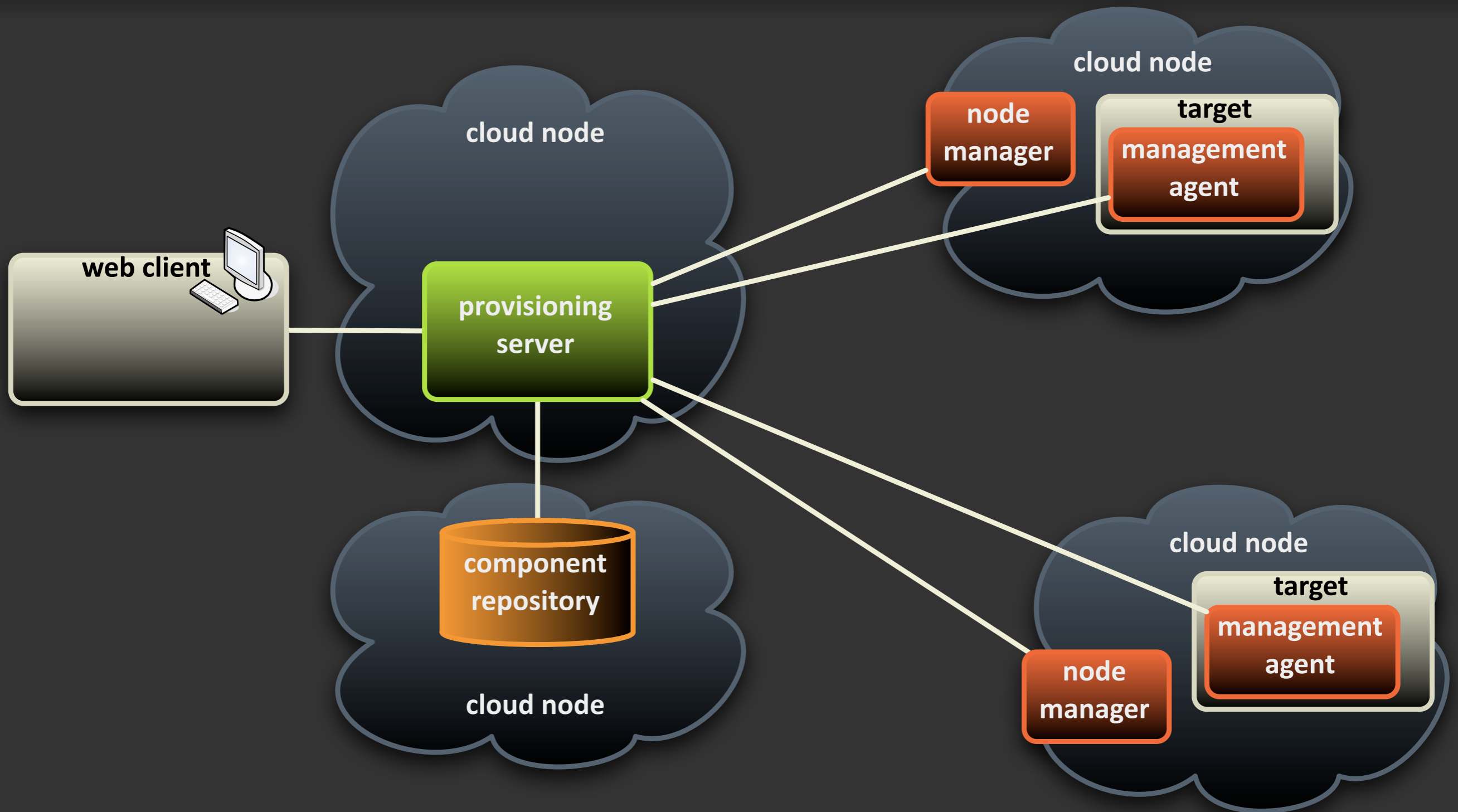
```
17:34 Checked for updates. none found
1 23:20 Bundle 23 stopped
1 2 13:23 Target started
2 2 13:24 Starting update from version 5 to 8
2 0 13:24 Bundle 37 updated
0 13:25 Update to version 8 succeeded
14:25 Target stopped
```

```
13:23 Target started
13:24 Starting update from version 5 to 8
13:24 Bundle 37 updated
13:25 Update to version 8 succeeded
14:25 Target stopped
```

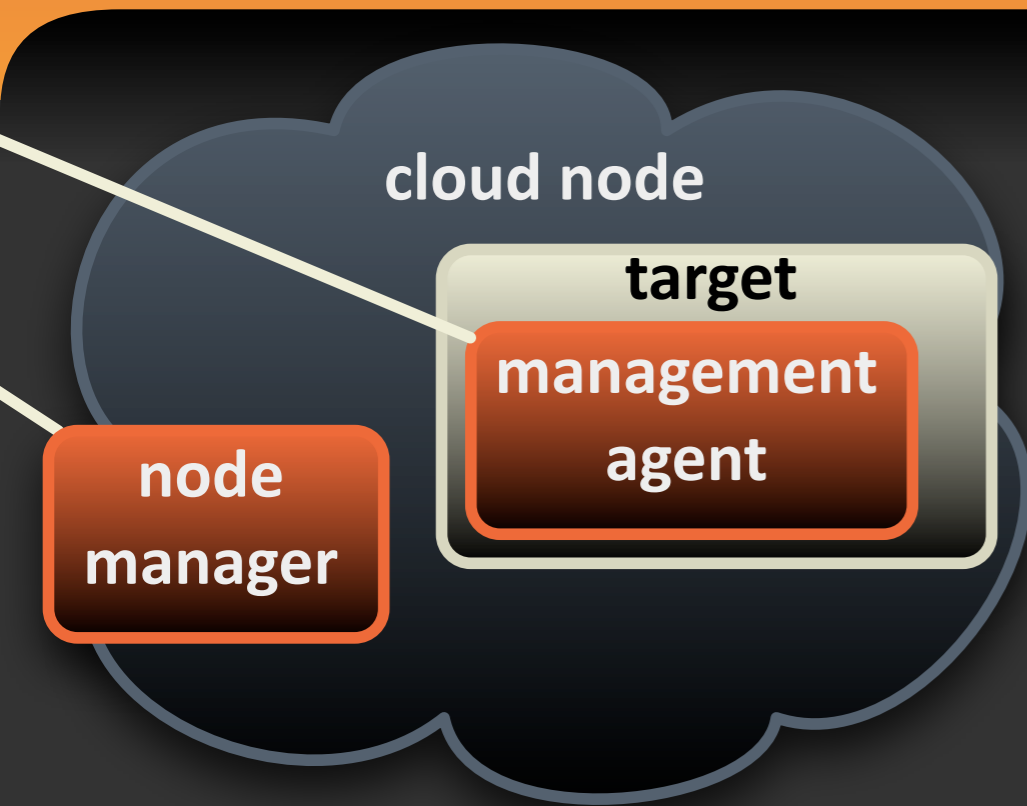
# ACE in the Cloud



# ACE in the Cloud



# Node Manager



- **Node Manager**
  - bootstraps the node
  - launches targets
  - measures performance data

# ACE UI Extensions

```
/**
 * Creates components for named extension points in the Vaadin UI. Extension factories
 * are used throughout the UI to allow other bundles to contribute features.
 */
public interface UIExtensionFactory {
    public static final String EXTENSION_POINT_KEY = "extension_point";
    public static final Object EXTENSION_POINT_VALUE_ARTIFACT = "artifact";
    public static final Object EXTENSION_POINT_VALUE_FEATURE = "feature";
    public static final Object EXTENSION_POINT_VALUE_DISTRIBUTION = "distribution";
    public static final Object EXTENSION_POINT_VALUE_TARGET = "target";

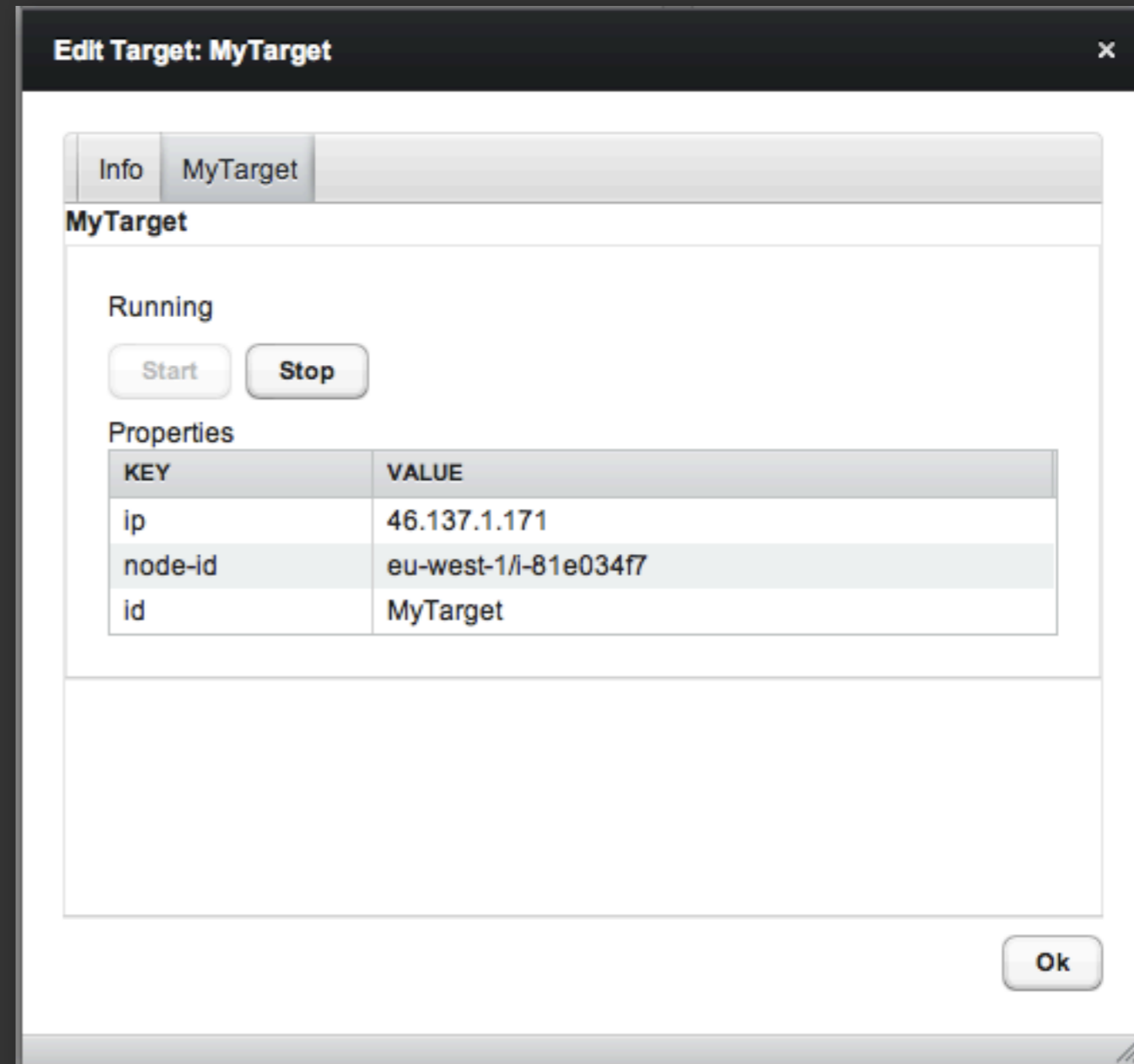
    /**
     * Creates a UI component for use in the extension point. The contents of the
     * context are extension-point dependent.
     */
    Component create(Map<String, Object> context);
}
```

# Cloud Extension

- **Currently supports one target per node**
- **Uses [jclouds.org](http://jclouds.org)**
- **Implementation for Amazon EC-2**

# Cloud Extension

- Currently supports one target per node
- Uses [jclouds.org](http://jclouds.org)
- Implementation for Amazon EC-2



# Hands On

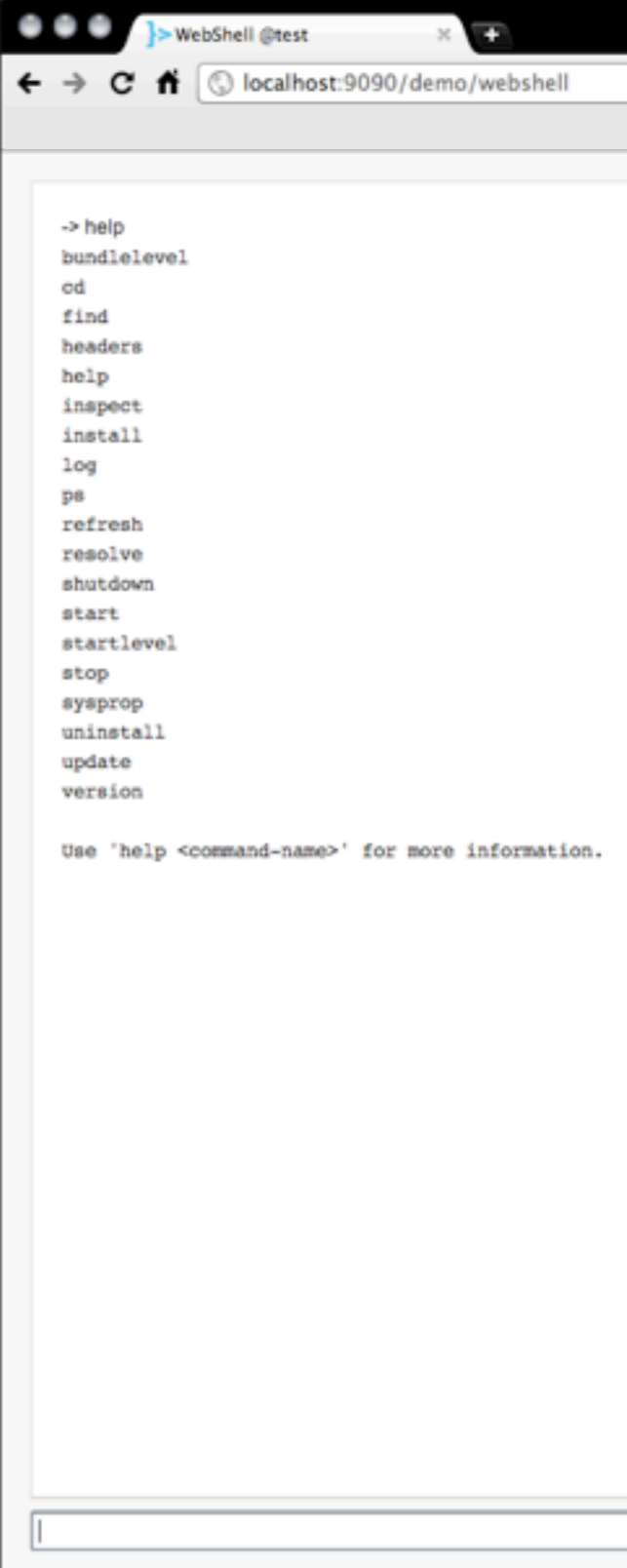
## 2. the Cloud

# Deploying ACE with ACE

- **For the next exercise we want to give everybody their own copy of ACE in the cloud**
- **We've pre-created most of them already, and will show you how to create and deploy a new instance**

# Exercise: “WebShell” in the cloud

- We’ve already created a ‘webshell/ deploy’ folder containing all relevant artifacts
- Using a pre-installed copy of ACE, we are going to deploy it on a new cloud node



The screenshot shows a web browser window with the title "WebShell @test" and the address bar "localhost:9090/demo/webshell". The main content area displays a list of commands for a webshell interface:

```
-> help
bundlelevel
cd
find
headers
help
inspect
install
log
ps
refresh
resolve
shutdown
start
startlevel
stop
sysprop
uninstall
update
version

Use 'help <command-name>' for more information.
```



# Hands On

## 3. ACE

# Building and running ACE

- **Checkout the sources:**

```
svn co http://svn.apache.org/repos/asf/incubator/ace/trunk/ ace
```

- **Build everything:**

```
cd ace  
mvn -DskipTests=true install
```

- **Launch the server:**

```
cd ace-target-devserver/  
cd target/  
cd org.apache.ace.target.devserver-0.8.0-SNAPSHOT-distribution/  
cd ace-devserver/  
sh run.sh
```

**on Windows:**

run.bat

- **Launch a target:**

```
cd ace-launcher  
cd target  
java -jar org.apache.ace.launcher-0.8.0-SNAPSHOT.jar
```

# Exercise

- After building ACE and running a server and a target, we can now locally deploy the “WebShell”

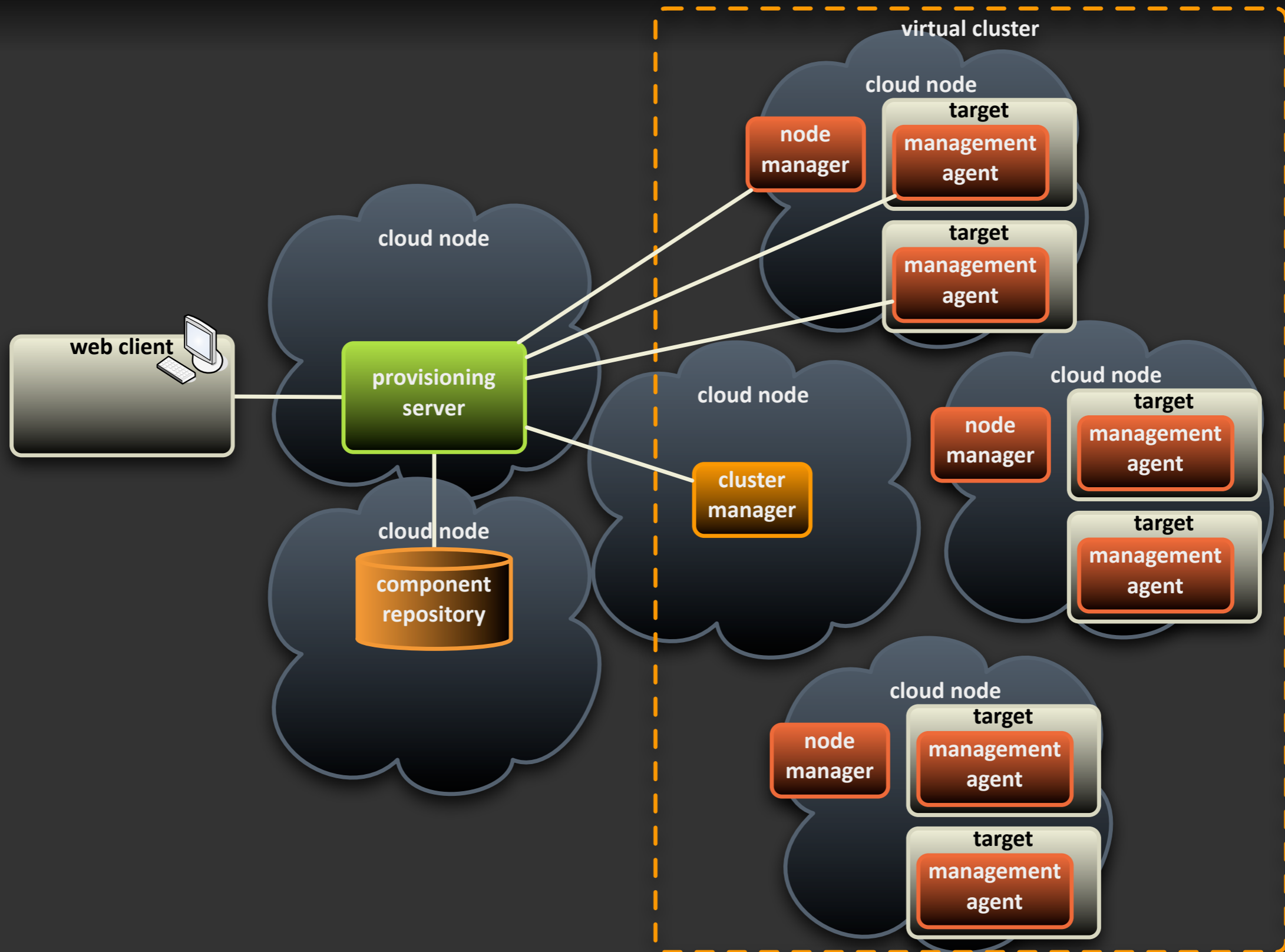


The screenshot shows a web browser window with the title "WebShell". The browser's address bar contains "lo". The main content area displays a list of commands available in the shell, including: help, bundlelevel, cd, find, headers, help, inspect, install, log, ps, refresh, resolve, shutdown, start, startlevel, stop, sysprop, uninstall, update, and version. At the bottom of the list, there is a prompt: "Use 'help <command>".

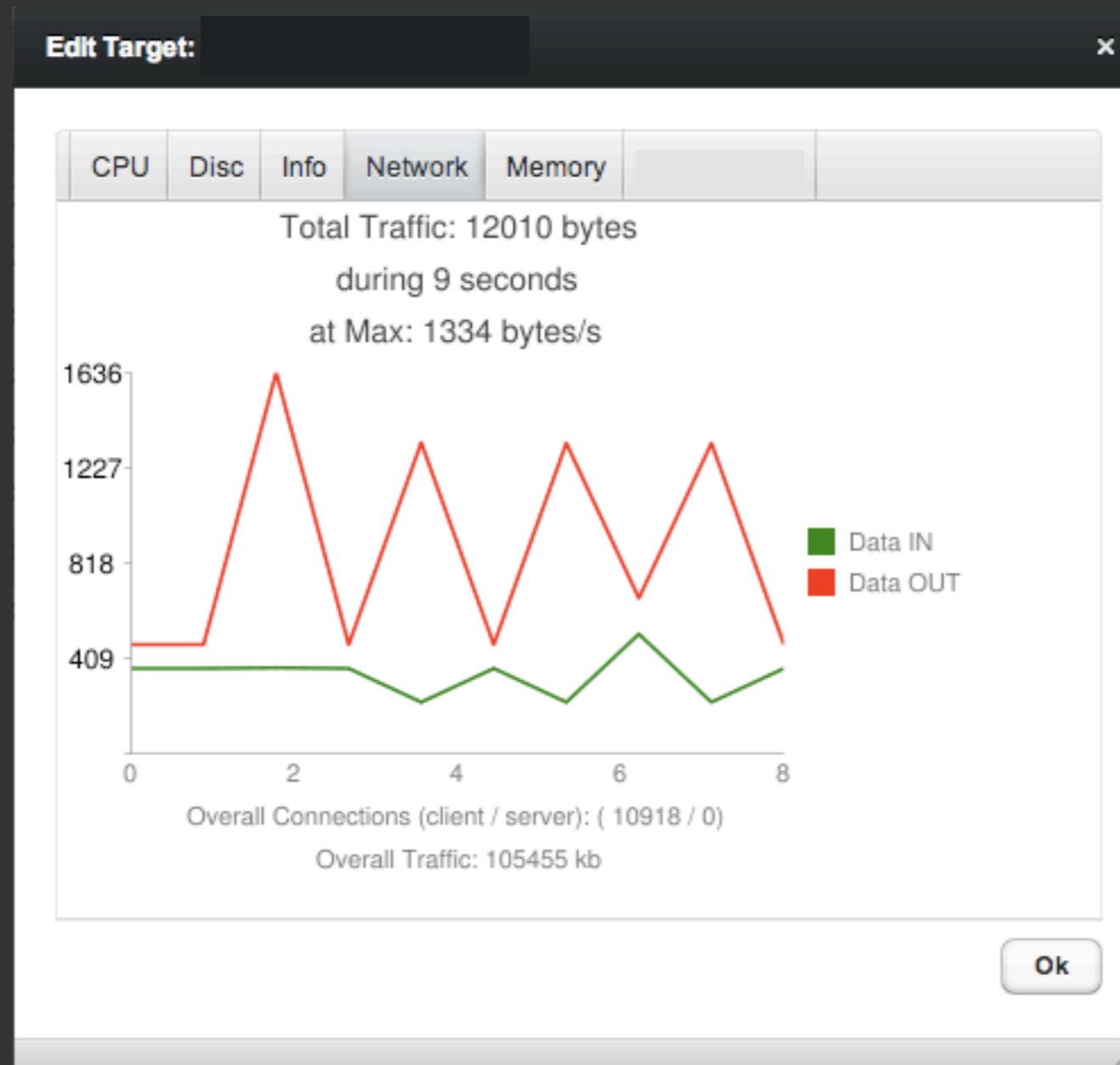
# Closing Remarks

- **We've looked at ACE, built it and ran it in the cloud and on our local system**
- **Brief peek into the future of cloud support in ACE**
- **Announcing a new open source project: Amdatu**

# Future of ACE in the Cloud



# Future of ACE in the Cloud





**amdatu**

# Amdatu

**“Amdatu is an Open Source platform and runtime for open, service oriented and cloud aware applications”**

- **Multi-tenancy**
- **Provisioning**
- **IaaS support**
- **(Elastic) Scalability**
- **Manageability**
- **Serviceability**
- **Reusability**



# Amdatu

**“Amdatu is an Open Source platform and runtime for open, service oriented and cloud aware applications”**

<http://amdatu.org/>

- **Multi-tenancy**
- **Provisioning**
- **IaaS support**
- **(Elastic) Scalability**
- **Manageability**
- **Serviceability**
- **Reusability**

# Amdatu Subprojects

- **Amdatu Core**
- **Amdatu Provisioning**
- **Amdatu Semantic**
- **Amdatu Auth**
- **Amdatu Social**
- **Amdatu Storage**
- **Amdatu Search**

# Links

- <http://incubator.apache.org/ace/>
- <http://felix.apache.org/>
- <http://amdatu.org/>
- <http://www.luminis.eu/?lang=en>
- <http://luminis-technologies.com/>