Jeremy Hughes – WebSphere Application Server product development

5th March 2014

DevOps & WebSphere: Experiences in *Chef* enabling the IBM WebSphere *Liberty* Profile

QCon

Please evaluate my talk via the mobile app!

Legal Disclaimer

- What IS WebSphere Liberty Profile ?
- What IS Chef ?
- Why use them together ?

WAS is the Java™ Foundation for IBM Software

Over 300 IBM offerings embed
or build upon WAS

Why WAS Liberty Profile?

- Ops use WAS for stability, scalability, performance …
- Dev less so. They have other requirements
  - Simple to configure
  - Small and fast
  - Quick to cycle through code, compile, test, debug in an IDE
  - Mac

⇒Variety of appservers used in dev

WAS Liberty Profile

WAS v8.5.5 Liberty Profile & WAS Developer Tools for Eclipse (WDT)

Dynamic Server Profile
Not static like Web Profile; configured by app at a fine-grained level

"Developer First" Focus
Simplified, shareable XML server config. New integrated messaging server, DynaCache support, new prog. models, such as Web Services, JMS & EJB-Lite.

Start fast, run efficiently
Starts in <3s; Mem footprint <50MB; (TradeLite benchmark)

Small Download
50MB for Web Profile features

Integrated tools
Powerful tools in WDT Eclipse feature. Enhanced for v8.5.5 prog models, Maven integration, ++

Dynamically Extensible
Install new features from repository (local or remote) with no svr restart

Web Profile Certified
Create web apps for the Java EE Web Profile standard.

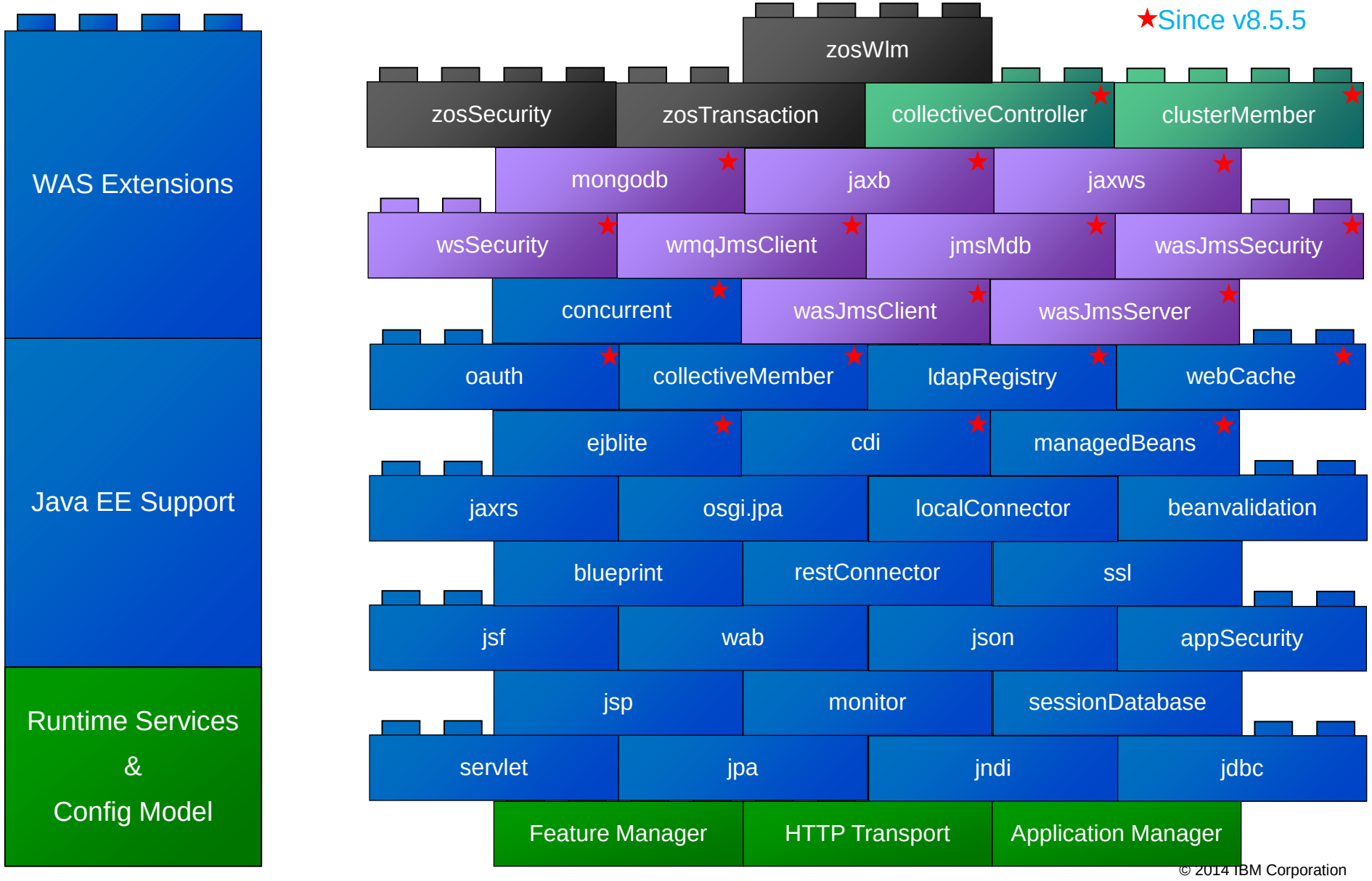
Lightweight cluster Mgmt
Liberty servers can join a lightweight cluster for workload balancing and high availability

Unzip install and deploy
IM or unzip to install. New option to deploy "server package" of app + config + required subset of server runtime for highest density deploy

Liberty Extensions
Add custom features and integrate 3rd party components via Liberty extensions interface

Fidelity to full profile WAS
Same reliable containers & QOS. Develop on Liberty profile and deploy to Liberty or full-profile WAS

Highly composable runtime based on ‘features’



Simplified Server Configuration

```
<server>
    <featureManager>
        <feature>jsp-2.2</feature>
        <feature>jdbc-4.0</feature>
    </featureManager>

    <logging traceSpecification="webcontainer=all=enabled:*=info=enabled"/>

    <application name="tradelite" location="tradelite.war"/>

    <dataSource jndiName="jdbc/TradeDataSource">
        <properties.derby.embedded databaseName="${server.config.dir}/tradedb"/>
    </dataSource>
</server>
```

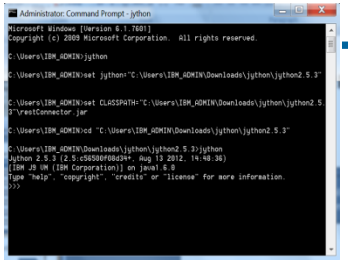
Features control which capabilities (bundles) are installed in the server

'singleton' configurations specify properties for a runtime service like logging

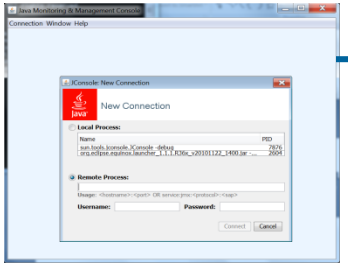
'instance' configurations specify multiple resources like applications and datasource definitions

Any of this configuration could be put into a separate xml file and 'included' in this 'master' configuration file

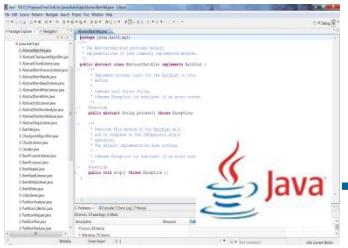
Control large sets of servers using Lightweight Liberty *Collectives and Clusters*



jython



jconsole



Java

Collective Controller

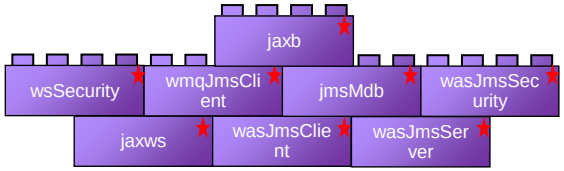
Liberty Profile (Collective Member)

Liberty Profile (Collective Member)

Liberty Cluster A

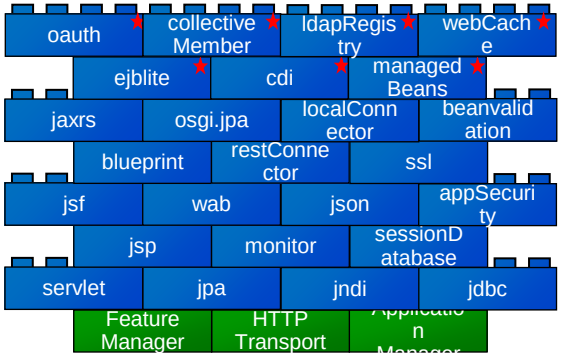
Liberty Cluster B

Size matters



Liberty **extended** archive 30MB

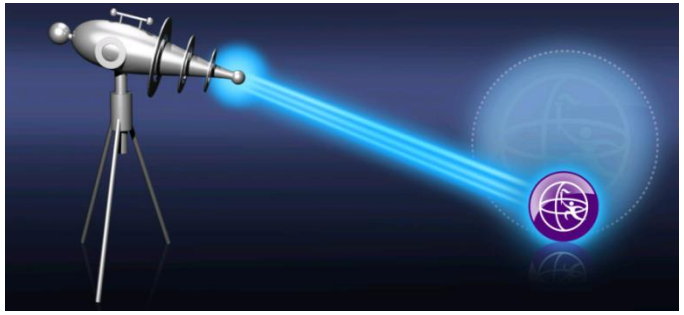
- **Install** only what you need to **minimize disk footprint.**
  - Modular “Archive Install” or
  - Standalone Installation Manager repository



Liberty **runtime** archive 50MB (Java EE Web Profile)

- **Start** only what you need - only configured features are started by the Liberty kernel to **minimize memory footprint**

- **Package** only what you need to **minimize a packaged server:**
  `server –package serverName` **`--include=minify`**



Starting out

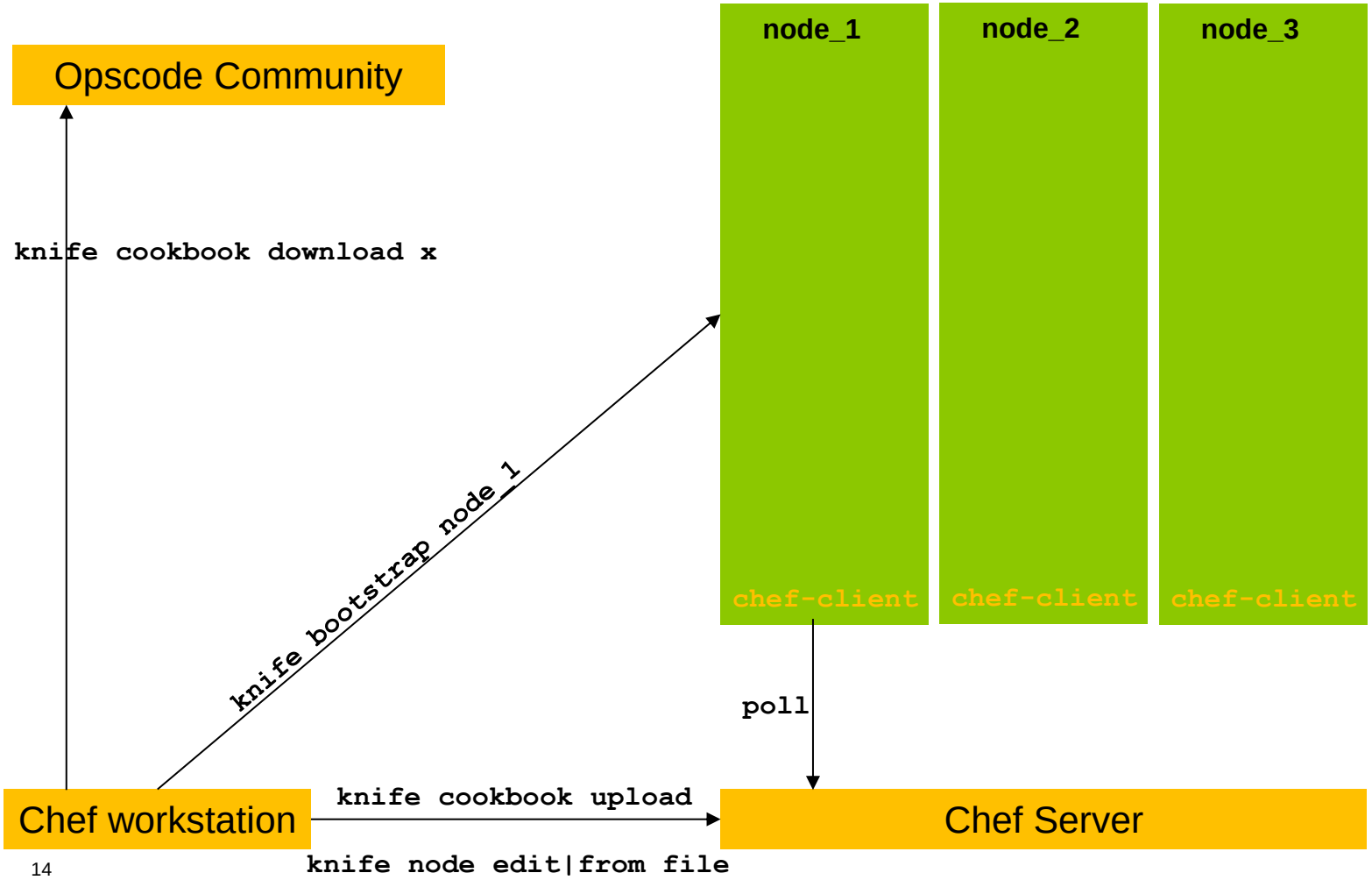
- June 2013 Chef Fundamentals education
  - 768 slides. On site, 1 week.
- 30 @ IBM … 2 sites plus some virtual; 3 days; 5 projects
- Mix of Devs, and Ops people attending
- Variety of projects
  - Internal deployment
  - SaaS / PaaS projects
  - additional chef-client platform support
  - **WAS Liberty & Chef**
- Starting out with Low->No Ruby skill
- Opscode prototypes

- Build out cookbooks in GitHub based on WAS Liberty Profile v8.5.5
- Advice from Opscode: You won't get it right the first time!

Chef

- Reproducible environments – tear it down, build it up => how long?
- Adding nodes => how much change => how long?
- Avoid the snowflake server
- Golden images … NO! Heavy, hard to transport, hard to mould
  - E.g. move all ssh ports to 2022
- Infrastructure as Code
  - Treat like any other code
  - Reconstruct business from code repo, data backup & bare metal

Simplified Chef node setup



Install & create

- **wlp** cookbook
  github.com/WASdev/ci.chef.wlp
- Install Liberty: archive|zip
  – IBM Licensing!
- Create server/s
  – Separate wlp/usr dir
- Set JVM options
- Manipulate server's live server.xml as hash
- Create init.d service wlp-<server_name>

Chef workstation

Chef Server

In development: chef-repo – a command line experience

```
chef-repo/
├── certificates
├── config
├── cookbooks
│   ├── application
│   ├── application_wlp
│   ├── java
│   ├── wlp
│   └── wlp-samples
├── data_bags
├── environments
└── roles
```

`knife cookbook site download java`

- Source @ github.com/WASdev repo name *vs* Chef Cookbook name
  - ci.chef.wlp -> wlp
  - ci.chef.wlp.application -> application_wlp
  - ci.chef.wlp.samples -> wlp-samples
  - ci.chef.wxs -> wxs
- Releases @ community.opscode.com

wlp cookbook

```
wlp
├── attributes          [:wlp] user /group / image url
├── CHANGELOG.md
├── doc                 doc wrappers (embedded doc everywhere)
├── libraries
├── metadata.rb         version info
├── providers
├── Rakefile
├── recipes             install and create Liberty Profile server
├── resources
├── spec                ChefSpec / RSpec tests
├── templates           .erb Erebus files … init.d
└── test                test-kitchen
```

Templating

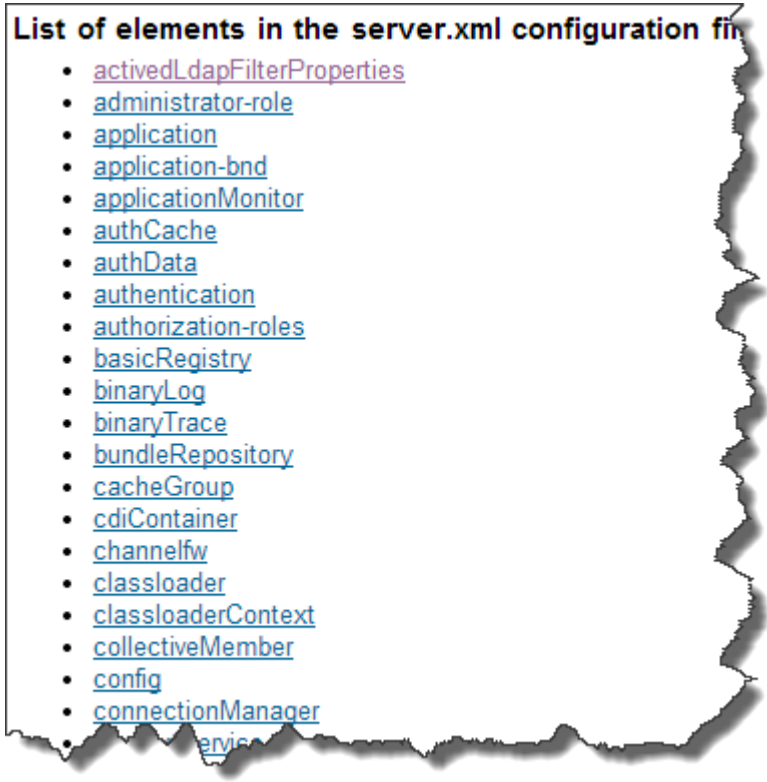
- Chef uses Erebus Ruby Gem for sophisticated ‘variable evaluation’
- Embedded Ruby code

```
#!/bin/sh

### BEGIN INIT INFO
# Provides:          wlp-<%= @serverName %>
# Required-Start:    $local_fs $remote_fs $network
# Required-Stop:     $local_fs $remote_fs $network
# Default-Start:     2 3 4 5
# Default-Stop:      0 1 6
# Short-Description: Start/Stop WebSphere Application Server Liberty Profile
### END INIT INFO

WLP_USER="<%=node['wlp']['user']%>"

if [ `id -un` != "$WLP_USER" ]; then
  exec su - $WLP_USER -- $0 "$@"
fi
```

Modelling server.xml … and <include>s

- Simple, minimal, but rich (!!)
  - 94 different element types
- Need to represent server config in code
- And keep ‘application’ cookbook’s deployment idiom

1. Manage raw server.xml
   - Clash with ‘application’ cookbook’s deployment idiom
2. server.xml.erb template?
   - Need complete model of elements and attributes in Ruby to manipulate them … onerous
   - Preserving includes during changes?
3. Simple, good enough:
   - Server config hash -> server.xml


List of elements in the server.xml configuration fi

- actived LdapFilterProperties
- administrator-role
- application
- application-bnd
- applicationMonitor
- authCache
- authData
- authentication
- authorization-roles
- basicRegistry
- binaryLog
- binaryTrace
- bundleRepository
- cacheGroup
- cdiContainer
- channelfw
- classloader
- classloaderContext
- collectiveMember
- config
- connectionManager

Hash to XML

```
default[:wlp][:servers][:defaultServer] = {
  "enabled" => true,
  "serverName" => "defaultServer",
  "description" => "Default Server",
  "featureManager" => {
    "feature" => [ "jsp-2.2" ]
  },
  "httpEndpoint" => {
    "id" => "defaultHttpEndpoint",
    "host" => "*",
    "httpPort" => "9080",
    "httpsPort" => "9443"
  }
}
```

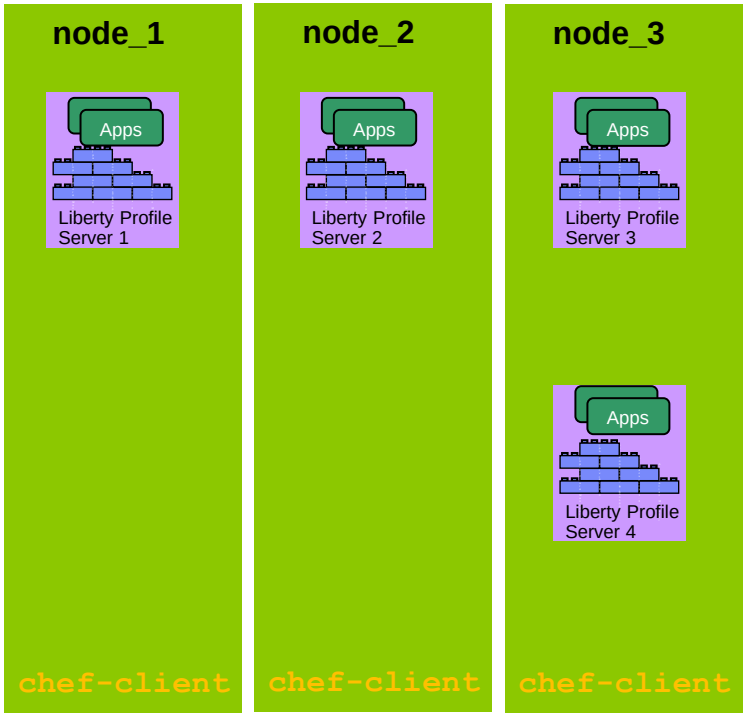
Converted by wlp cookbook

```
<server description="Default Server">
  <featureManager>
    <feature>jsp-2.2</feature>
  </featureManager>
  <httpEndpoint id="defaultHttpEndpoint" host="*" httpPort="9080" httpsPort="9443"/>
</server>
```

Deploy application

- **application_wlp** cookbook
  github.com/WASdev/ci.chef.wlp.application
- Extends Opscode's 'application'
  - Template Method pattern
  - before_compile (server install)
  - before_deploy (copy app & include.xml)
- Deploy application of any type
  .ear | .war | .eba

```
application "my-app" do
  path "/usr/local/my-app"
  repository "/nas/distro/my-app.war"
  revision "..."
  scm_provider Chef::Provider::File::Deploy

  wlp_application do
    server_name "MyAppServer"
    features [ "jsp-2.2", "servlet-3.0" ]
  end
end
```



Chef workstation

Chef Server

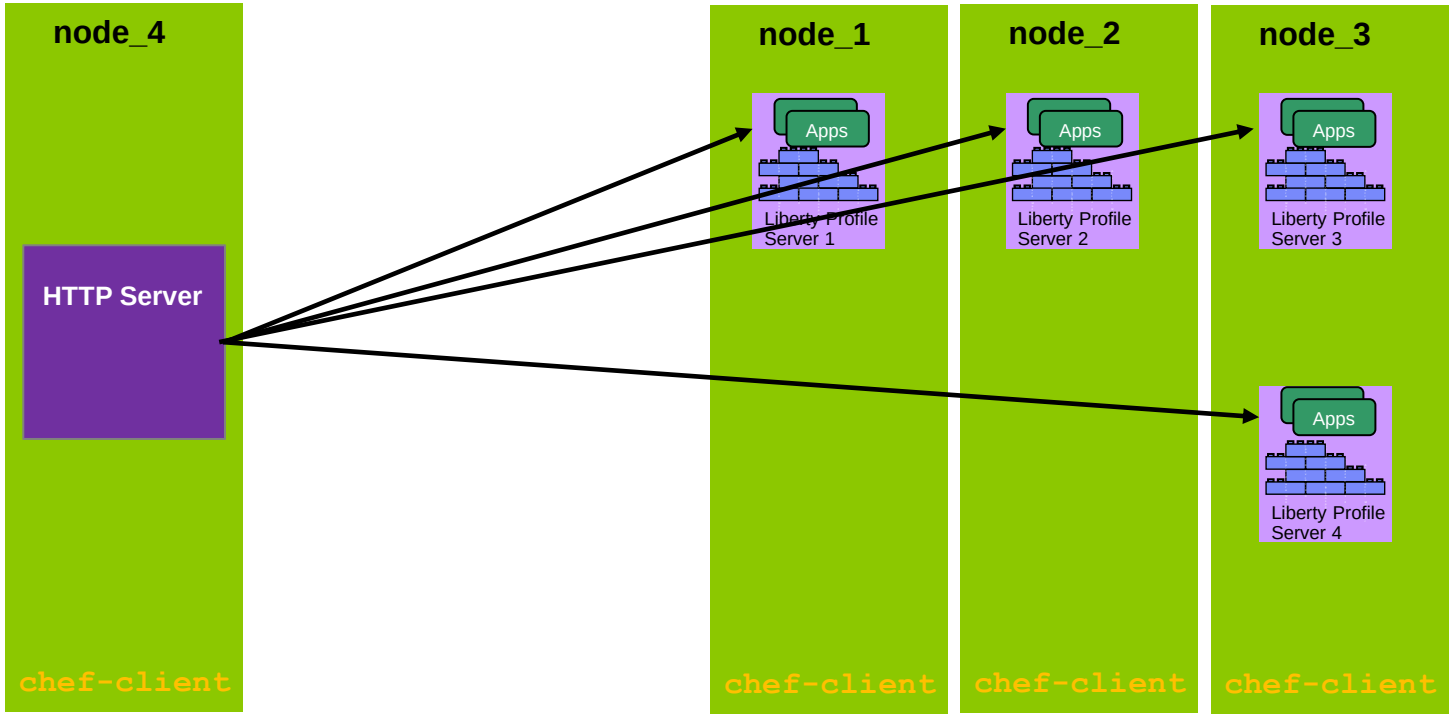
Release 0.1

- Tests!
  - foodcritic for lint
  - ChefSpec for unit tests
  - test-kitchen for integration tests with Vagrant + VirtualBox
  - Multiple Chef versions and platforms
- Team development needs Maintainer ID
  - Individuals marked as collaborators
  - Anyone can do the ‘share’
- How?
  - Update CHANGELOG.md

```
$ knife cookbook share wlp –u ibm-was
$ git tag -a 0.1 -m "0.1 release"
$ git push origin master –tags
```

  - Blog http://bit.ly/1i3rv2Y Tweet and repeat

Cookbook combinations: load balancing



Chef workstation

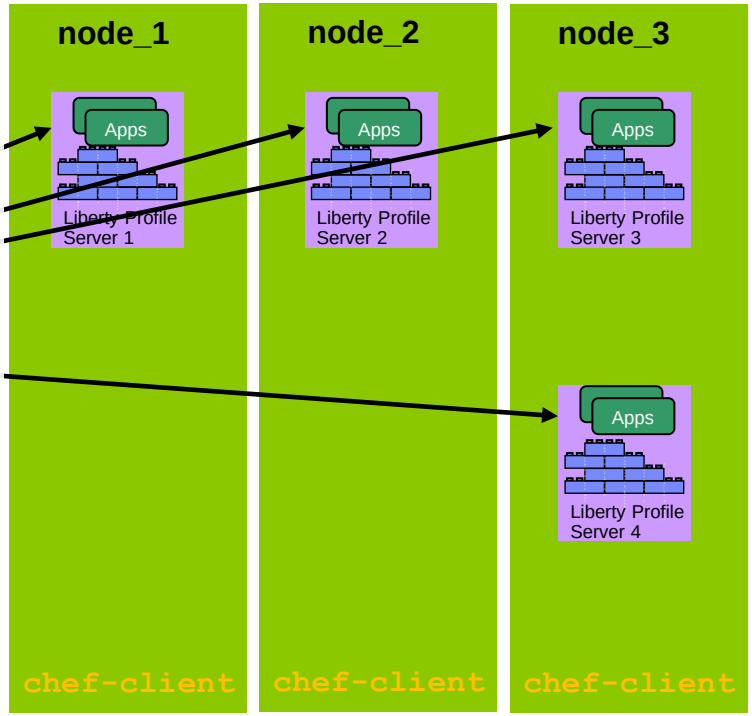
Chef Server

Cookbook combinations: load balancing sample

- Combine wlp cookbooks with apache2
  github.com/WASdev/ci.chef.wlp.samples
- httpd.conf.erb

```
<Proxy balancer://mycluster>

        <% @ipaddresses.each do |ip| -%>
        BalancerMember http://<%= ip %>:9080
        <% end -%>
```

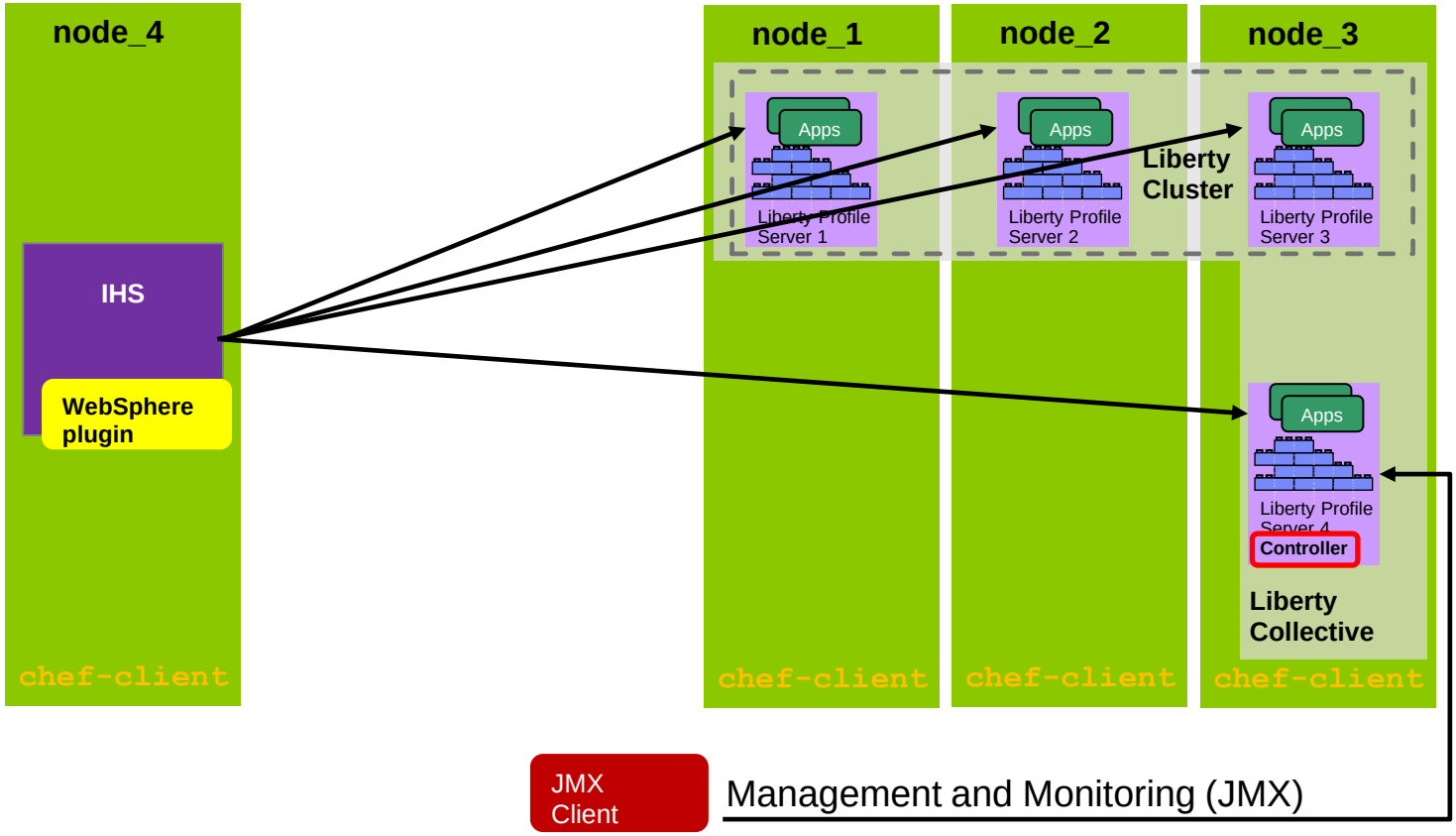
- Search Chef Server metadata for nodes in a specific cluster role



Chef workstation

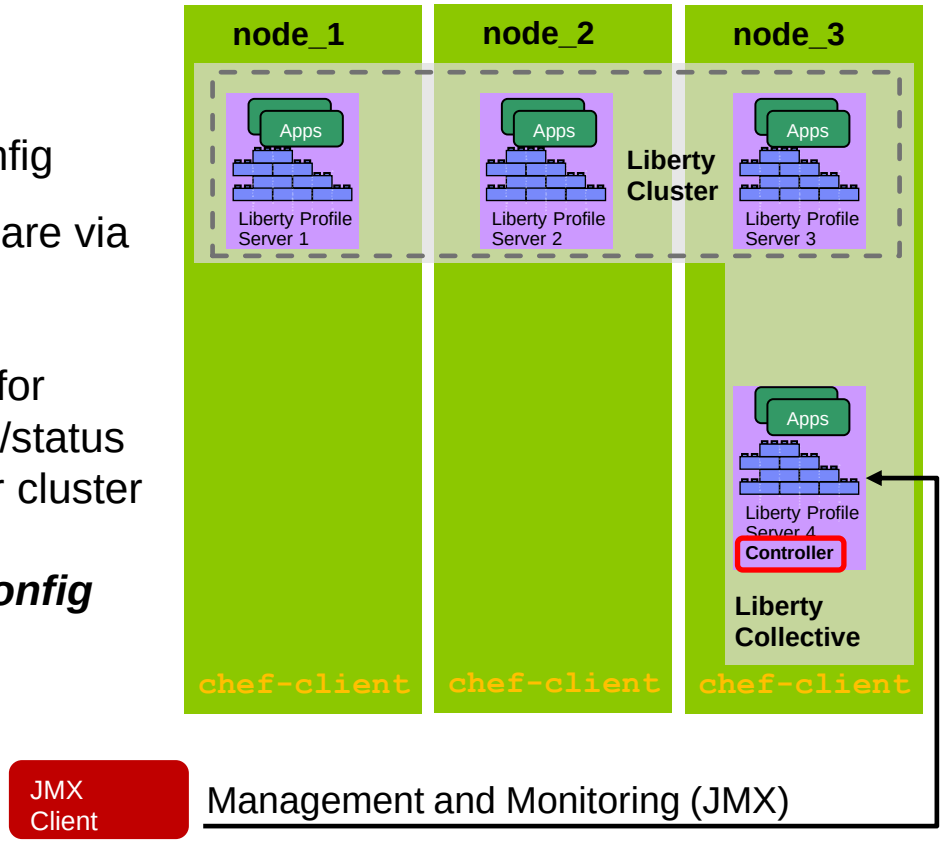
Chef Server

Clusters & Collectives. Liberty config management at scale.

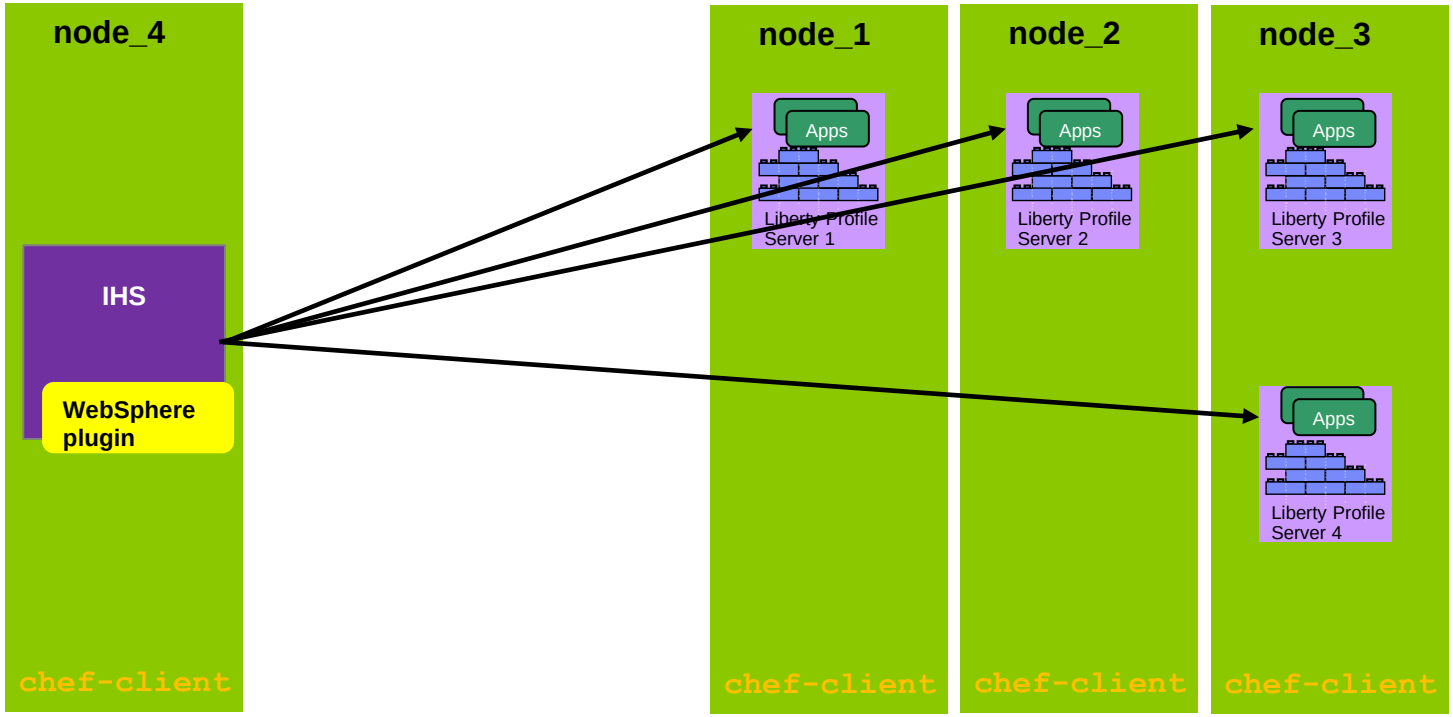


Clusters & Collectives. Liberty config management at scale.

- Access all members through *Controller*
- A server member owns its config
- JMX based management ops are via collective controller instead of individual server
  - ServerCommandsMBean for individual server start/stop/status
  - ClusterManagerMBean for cluster start/stop/status & ***generateClusterPluginConfig***
  - Controller uses ssh



IBM HTTP Server (I.H.S.)



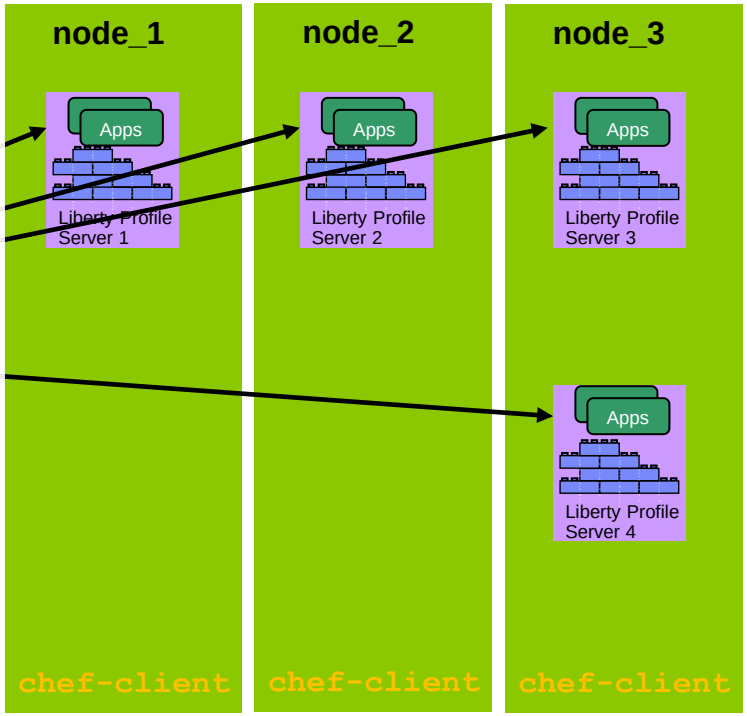
Chef workstation

Chef Server

IBM HTTP Server (I.H.S.)

- IBM products use Installation Manager
- First install IM
- Use IM to install I.H.S.
- IM Cookbook (not yet released)

  github.com/WASdev/ci.chef.installationmanager
- WebSphere Plugin config can be collected from each node in role
- For standalone sets of servers behind I.H.S. where no aggregated JMX required
  => Chef is good option
- Caveat: ongoing work!



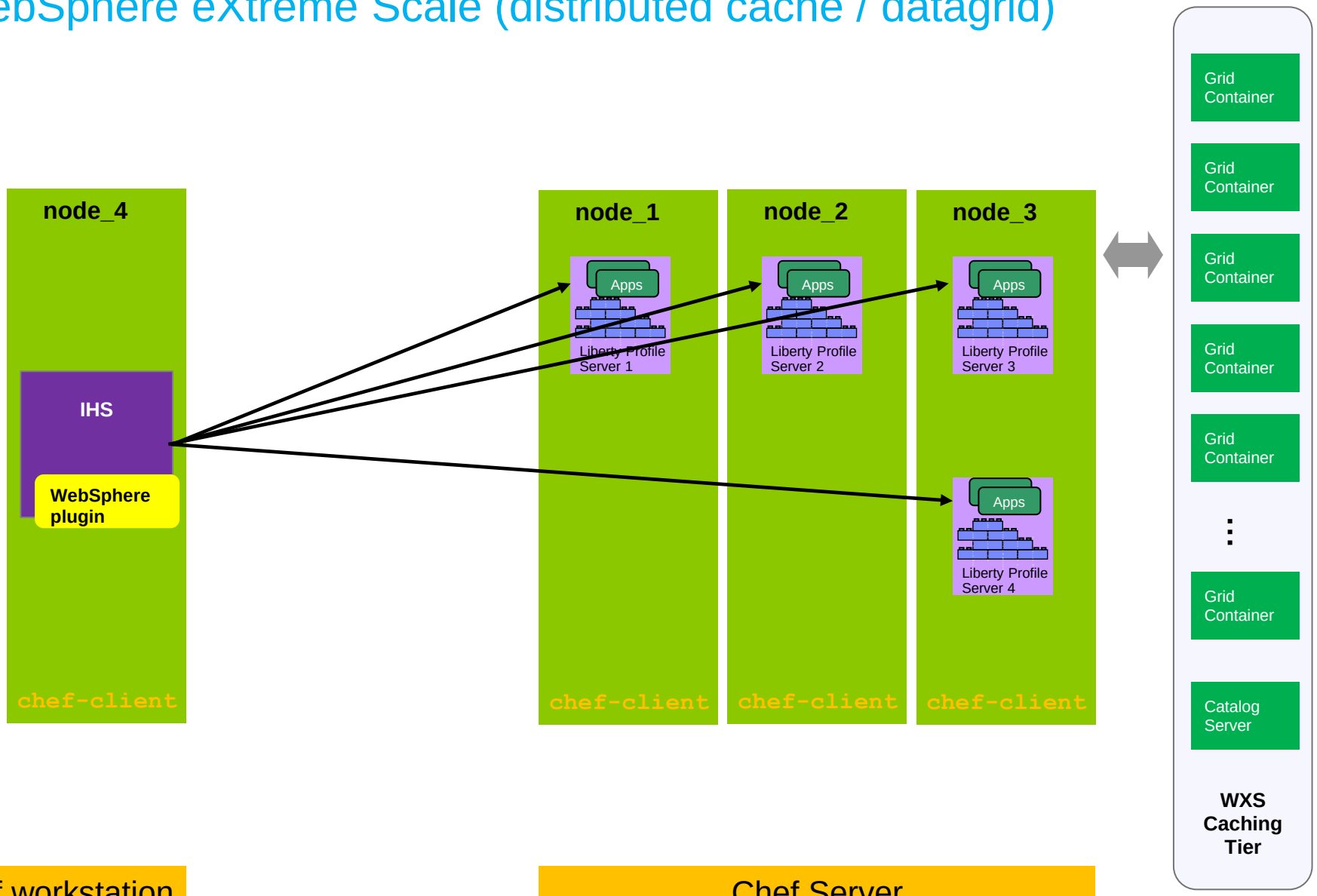
Chef workstation

Chef Server

Release 0.2 & 0.2.1

- Resource for installing individual features (.esa files)
  - User features
  - Liberty Repository features
- Encoding passwords
  - Utility to generate aes passwords for inclusion in server.xml
- Java cookbook install issues for IBM Java SDK
  - SDK .bin distro: problem on Ubuntu – RPM (!!!)
  - SDK .tar.gz: /etc/alternatives & USER_INSTALL_DIR
- Share, Blog http://bit.ly/1dYUUZo Tweet and repeat

WebSphere eXtreme Scale (distributed cache / datagrid)

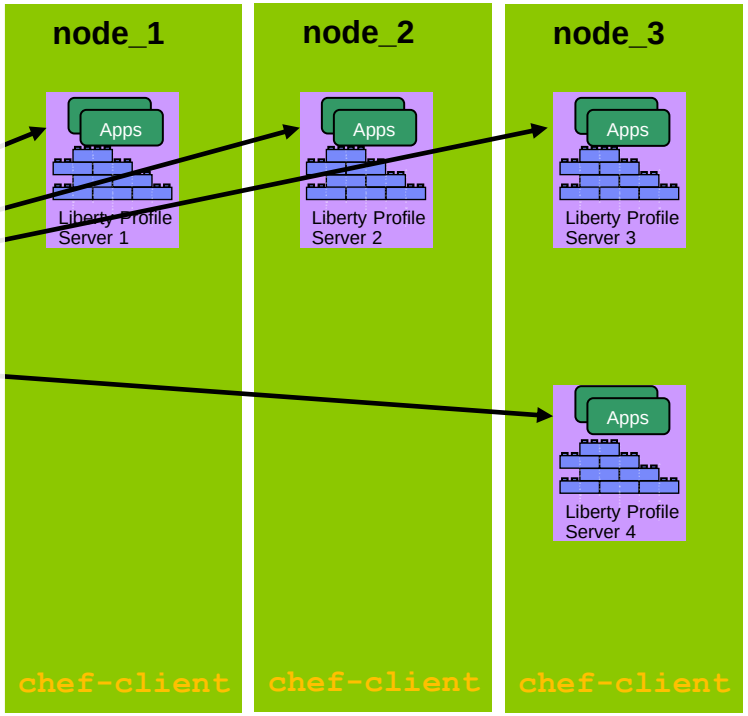


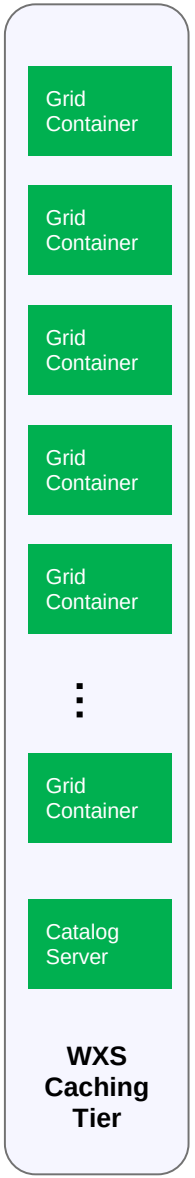
WebSphere eXtreme Scale (distributed cache / datagrid)

- WXS Cookbook (not yet released)

  github.com/WASdev/ci.chef.wxs

- Liberty features for
  - application clients
  - eXtreme Scale server components
- Caveat: ongoing work!





Chef workstation

Chef Server

Directed & convergent styles

- Agents everywhere
- Convergent
  - Patches & updates associated with ‘role’
  - Agents are in ‘role’
  - Ensures online machines will have the right stuff (related to poll timeout)
- Directed
  - Ordered set of changes e.g. DB update before application update
  - Keep track of ‘what should be’ and ‘what is’ on a target & progress
  - Multi-tier deployments
- Complementary (but with overlap)

https://www.ibmdw.net/urbancode/2012/09/05/convergent-vs-directed-deployments

Investing in Release Automation … Introducing UrbanCode

Enabling clients to more rapidly deliver mobile, cloud, big data and traditional applications with high quality and low risk

Drive down cost

Remove manual effort and wasted resource time with push button deployment processes

Speed time to market

Simple, graphical process designer, with built-in actions to quickly create deployment automation

Reduce risk

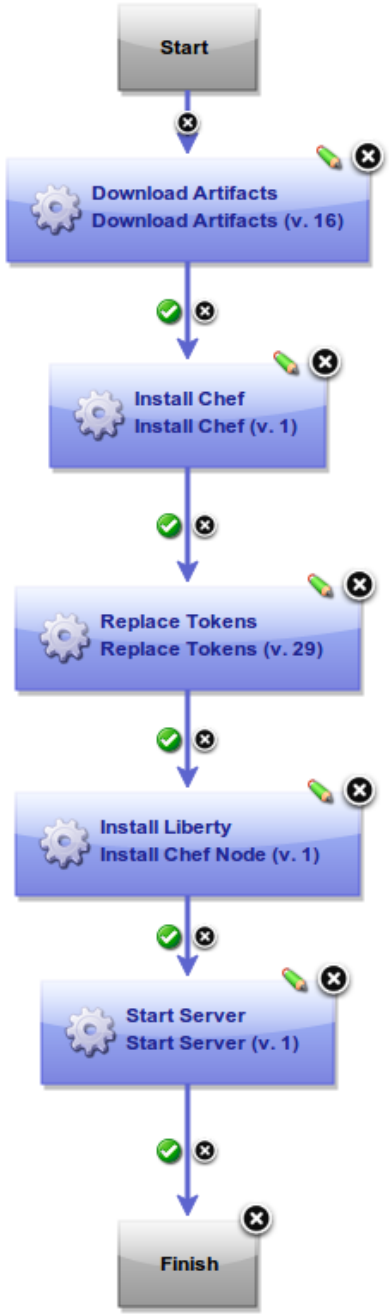
Robust configuration management, coordinated release processes, audits, and traceability

IBM UrbanCode Deploy orchestrates and automates the deployment of applications, databases and configurations into development, test and production environments, helping to drive down cost, speed time to market with reduced risk.

IBM UrbanCode Release is an intelligent collaboration release management solution that replaces error-prone manual spreadsheets and streamlines release activities for application and infrastructure changes.

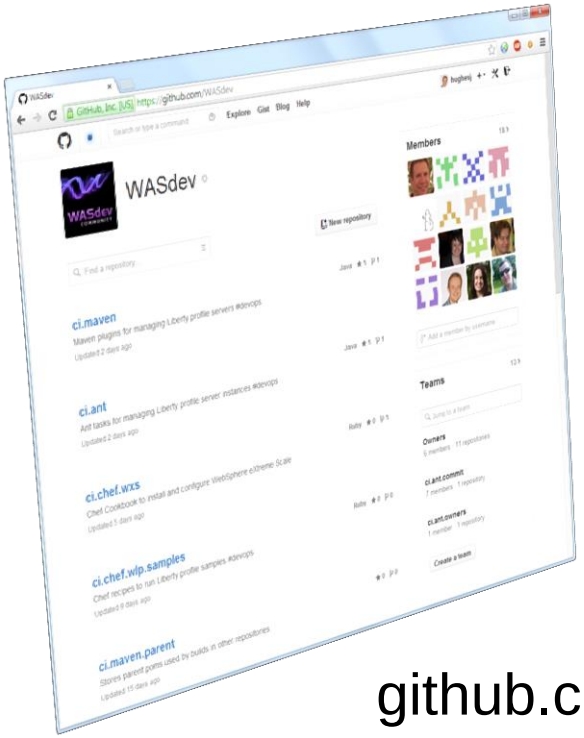
UrbanCode Deploy flow for Chef & Liberty

- Directed DevOps approach
- Automate deployment of applications and components
- Visual process designer
- Plug-in steps for application containers, web servers, network devices, database deployment etc
  www.ibmdw.net/urbancode/plugins/
- Chef Plugin
- WebSphere Liberty Profile plugin (install & create)

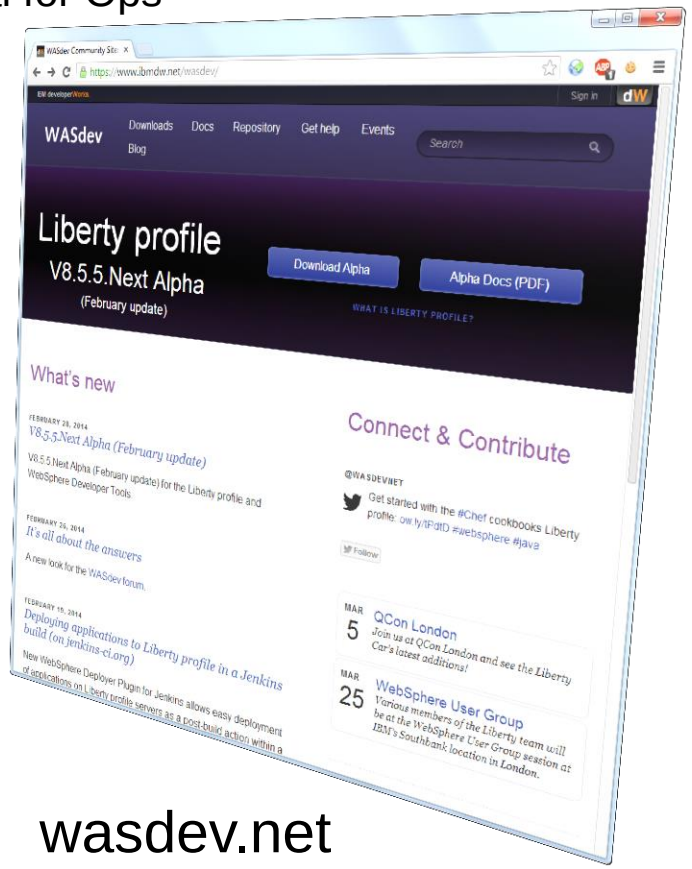


Chef & WebSphere Liberty Profile: no-charge downloads

- Chef is Infrastructure as Code: model your infrastructure as code. www.getchef.com
- WebSphere Liberty Profile: designed for Developers, ideal for Ops
- Visit us at WASdev and our GitHub org:



github.com/WASdev



wasdev.net

QCon

Please evaluate my talk via the mobile app!