RFC 240 - Compute Management Service

Draft

10 Pages

Abstract

A large variety of ways exist today to create, run and manage computing instances, whether this be in the cloud, on a computing grid or on a container-based environment. While physically highly diverse, managing these on a logical level can be seen as similar. This RFP seeks to define a common service to facilitate managing compute nodes across a wide variety of platforms.

0 Document Information

0.1 License

DISTRIBUTION AND FEEDBACK LICENSE, Version 2.0

The OSGi Alliance hereby grants you a limited copyright license to copy and display this document (the “Distribution”) in any medium without fee or royalty. This Distribution license is exclusively for the purpose of reviewing and providing feedback to the OSGi Alliance. You agree not to modify the Distribution in any way and further agree to not participate in any way in the making of derivative works thereof, other than as a necessary result of reviewing and providing feedback to the Distribution. You also agree to cause this notice, along with the accompanying consent, to be included on all copies (or portions thereof) of the Distribution. The OSGi Alliance also grants you a perpetual, non-exclusive, worldwide, fully paid-up, royalty free, limited license (without the right to sublicense) under any applicable copyrights, to create and/or distribute an implementation of the Distribution that: (i) fully implements the Distribution including all its required interfaces and functionality; (ii) does not modify, subset, superset or otherwise extend the OSGi Name Space, or include any public or protected packages, classes, Java interfaces, fields or methods within the OSGi Name Space other than those required and authorized by the Distribution. An implementation that does not satisfy limitations (i)-(ii) is not considered an implementation of the Distribution, does not receive the benefits of this license, and must not be described as an implementation of the Distribution. "OSGi Name Space" shall mean the public class or interface declarations whose names begin with "org.osgi" or any recognized successors or replacements thereof. The OSGi Alliance expressly reserves all rights not granted pursuant to these limited copyright licenses including termination of the license at will at any time.

EXCEPT FOR THE LIMITED COPYRIGHT LICENSES GRANTED ABOVE, THE OSGi ALLIANCE DOES NOT GRANT, EITHER EXPRESSLY OR IMPLIEDLY, A LICENSE TO ANY INTELLECTUAL PROPERTY IT, OR ANY THIRD PARTIES, OWN OR CONTROL. Title to the copyright in the Distribution will at all times remain with the OSGi Alliance. The example companies, organizations, products, domain names, e-mail addresses, logos, people, places, and events depicted therein are fictitious. No association with any real company, organization, product, domain name, email address, logo, person, place, or event is intended or should be inferred.

THE DISTRIBUTION IS PROVIDED "AS IS," AND THE OSGi ALLIANCE (INCLUDING ANY THIRD PARTIES THAT HAVE CONTRIBUTED TO THE DISTRIBUTION) MAKES NO REPRESENTATIONS OR WARRANTIES, EXPRESS OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, NON-INFRINGEMENT, OR TITLE; THAT THE CONTENTS OF THE DISTRIBUTION ARE SUITABLE FOR ANY PURPOSE; NOR THAT THE IMPLEMENTATION OF SUCH CONTENTS WILL NOT INFRINGE ANY THIRD PARTY PATENTS, COPYRIGHTS, TRADEMARKS OR OTHER RIGHTS.
NEITHER THE OSGi ALLIANCE NOR ANY THIRD PARTY WILL BE LIABLE FOR ANY DIRECT, INDIRECT, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF OR RELATING TO ANY USE OR DISTRIBUTION OF THE DISTRIBUTION.

Implementation of certain elements of this Distribution may be subject to third party intellectual property rights, including without limitation, patent rights (such a third party may or may not be a member of the OSGi Alliance). The OSGi Alliance is not responsible and shall not be held responsible in any manner for identifying or failing to identify any or all such third party intellectual property rights.

The Distribution is a draft. As a result, the final product may change substantially by the time of final publication, and you are cautioned against relying on the content of this Distribution. You are encouraged to update any implementation of the Distribution if and when such Distribution becomes a final specification.

The OSGi Alliance is willing to receive input, suggestions and other feedback (“Feedback”) on the Distribution. By providing such Feedback to the OSGi Alliance, you grant to the OSGi Alliance and all its Members a non-exclusive, non-transferable,

worldwide, perpetual, irrevocable, royalty-free copyright license to copy, publish, license, modify, sublicense or otherwise distribute and exploit your Feedback for any purpose. Likewise, if incorporation of your Feedback would cause an implementation of the Distribution, including as it may be modified, amended, or published at any point in the future ("Future Specification"), to necessarily infringe a patent or patent application that you own or control, you hereby commit to grant to all implementers of such Distribution or Future Specification an irrevocable, worldwide, sublicenseable, royalty free license under such patent or patent application to make, have made, use, sell, offer for sale, import and export products or services that implement such Distribution or Future Specification. You warrant that (a) to the best of your knowledge you have the right to provide this Feedback, and if you are providing Feedback on behalf of a company, you have the rights to provide Feedback on behalf of your company; (b) the Feedback is not confidential to you and does not violate the copyright or trade secret interests of another; and (c) to the best of your knowledge, use of the Feedback would not cause an implementation of the Distribution or a Future Specification to necessarily infringe any third-party patent or patent application known to you. You also acknowledge that the OSGi Alliance is not required to incorporate your Feedback into any version of the Distribution or a Future Specification.

I HEREBY ACKNOWLEDGE AND AGREE TO THE TERMS AND CONDITIONS DELINEATED ABOVE.

0.2 Trademarks

OSGi™ is a trademark, registered trademark, or service mark of the OSGi Alliance in the US and other countries. Java is a trademark, registered trademark, or service mark of Oracle Corporation in the US and other countries. All other trademarks, registered trademarks, or service marks used in this document are the property of their respective owners and are hereby recognized.

0.3 Feedback

This document can be downloaded from the OSGi Alliance design repository at https://github.com/osgi/design The public can provide feedback about this document by opening a bug at https://www.osgi.org/bugzilla/.

0.4 Table of Contents

0.5 Terminology and Document Conventions

The key words "MUST", "MUST NOT", "REQUIRED", "SHALL", "SHALL NOT", "SHOULD", "SHOULD NOT", "RECOMMENDED", "NOT RECOMMENDED", "MAY" and "OPTIONAL" in this document are to be interpreted as described in 10.1.

`Source code is shown in this typeface.`

0.6 Revision History

The last named individual in this history is currently responsible for this document.

Revision	Date	Comments
Initial	April 2018	Initial version based on RFP 179.

1 Introduction

OSGi RFP 133 (http://osgi.org/bugzilla/show_bug.cgi?id=114) provides the foundation requirements of many cloud-related activities in the OSGi Alliance. It includes requirements around management of nodes in a cloud environment where a node runs an OSGi framework. This RFP seeks to elaborate on the cloud environment management requirements, expanding them to more general container-based environments as well as adding support for nodes that can run any type of application, such as a database or load balancer.

2 Application Domain

The domain of this RFP is Cloud computing, Container-based computing and Grid computing. Effectively any environment where compute nodes or instances can be created programmatically. The RFP seeks to define requirements for a common API specification to manage Compute nodes across these environments.

Existing solutions in this area include, but are not limited to:

- Apache JClouds : http://jclouds.apache.org
- OpenStack: http://www.openstack.org
- Cloud Foundry: https://www.cloudfoundry.org
- TOSCA: http://docs.oasis-open.org/tosca/
- CAMP: https://www.oasis-open.org/committees/tc_home.php?wg_abbrev=camp
- Open Container Initiative: https://www.opencontainers.org/

2.1 Terminology + Abbreviations

Compute Node – an entity that can execute code. Either a physical computer, virtual machine or lightweight container. The entity provides an operating system, such as Microsoft Windows, Linux, other unix or similar. The node is connected to a IP-based network.

3 Problem Description

Many cloud vendors exist today that provide the capability to create compute nodes allowing the user to run applications on these vendor-provided environments. Compute nodes can generally be created using a remote API, web interface or command-line tool.

More recently container-based computing is becoming popular, docker is currently dominant in this space but other solutions such as rkt (Rocket) are also emerging. With container-based computing not an entire Virtual Machine is created, but rather a container to run a single program is created. While containers are isolated, the operating system and baseline functionality such as programming language support is often shared with other containers running on the same physical or virtual underlying platform.

While some standardization is happening on the virtual machine level, e.g. openstack, these standards don't apply to container based solutions. In the Java world some opensource projects exist that try to address this

issue, such as Apache jclouds, however the biggest issue here is that as jclouds provides its own API it also needs to provide all of the bindings to the technologies. This means that jclouds supports an enormous amount of technologies, where not all of these are equally well maintained.

By defining an OSGi service API, implementors can decide to only support one target: the one they care about. Multiple targets can be supported at runtime by combining multiple API implementors, each of which will be represented, for example, by a service in the service registry. Having a specification-defined API supports this federated approach.

4 Requirements

CM0010 – The solution must provide a mechanism to create and destroy compute nodes.

CM0020 – It must be possible to implement the solution for existing compute platforms, including, but not limited to, cloud-based platforms, grid computing-based platforms and/or container-based platforms.

CM0030 – The solution must provide a mechanism to specify the root image to run on the compute node.

CM0040 – The solution must provide a mechanism to specify compute node parameters such as the amount of memory, cpu and exposed network ports.

CM0050 – The solution must provide a means to specify environment variables to be set in the compute node.

CM0060 – The solution must provide a mechanism to run an executable in the compute node once available.

CM0070 – The solution must support listing and querying of compute nodes.

CM0080 – The solution must support specifying that more than one node of a given definition be created.

CM0090 – The solution must allow the number of requested nodes of a given definition to be changed after the initial launch of the associated compute node(s).

CM0100 – The solution must support assigning logical names to nodes that can be used to address them.

CM0110 – The solution must allow integration with security features offered by the compute platform.

CM0120 – The solution must provide support for load balancers across a cluster of nodes.

CM0130 – The solution must provide the external address of a load balancer, if available.

CM0140 – The solution must support addressing a load balancer with a logical name.

CM0150 – The solution must provide runtime management information, for example via DTOs.

CM0160 – The solution must allow the effects of the service to follow a different lifecycle than the runtime of Compute Management Service.

CM0170 – The solution must be able to discover compute nodes created during previous runs of the Compute Management Service.

CM0180 – It must be possible to provide secrets to the computer node in a secure way

CM0190 – It must be possible to specify whether compute nodes have a private or public IP address or both.

CM0200 – It must be possible to create compute nodes which are not directly accessible by the creator, such as nodes that are connected to a different network than the creator.

4.1 Persistent Volumes

PE0010 – The solution must be able to create, list, and delete persistent volumes like Amazon Elastic Storage Blocks or Google's Persistent volumes

PE0020 – It must be possible to attach a persistent volume to a compute node when the node is created

PE0030 – It must be possible to have a unique DNS name for a compute node that is based on the attached persistent volume name.

4.2 Notifications

NO0010 – It must be possible to get notifications when nodes are created, are stopped, or die

NO0020 – It must be possible to receive notifications of the various states that a compute node goes through, including the following states: CREATED, WAITING, PENDING, STARTING, ACTIVE, STOPPING, STOPPED, FAILED. Backing platforms may not support all states. Implementations are not required to provide notifications for states not supported by the backing platform.

5 Technical Solution

First give an architectural overview of the solution so the reader is gently introduced in the solution (Javadoc is not considered gently). What are the different modules? How do the modules relate? How do they interact? Where do they come from? This section should contain a class diagram. Then describe the different modules in detail. This should contain descriptions, Java code, UML class diagrams, state diagrams and interaction diagrams. This section should be sufficient to implement the solution assuming a skilled person.

Strictly use the terminology a defined in the Problem Context.

On each level, list the limitations of the solutions and any rationales for design decisions. Almost every decision is a trade off so explain what those trade offs are and why a specific trade off is made.

Address what security mechanisms are implemented and how they should be used.

6 Data Transfer Objects

RFC 185 defines Data Transfer Objects as a generic means for management solutions to interact with runtime entities in an OSGi Framework. DTOs provides a common, easily serializable representation of the technology.

For all new functionality added to the OSGi Framework the question should be asked: would this feature benefit from a DTO? The expectation is that in most cases it would.

The DTOs for the design in this RFC should be described here and if there are no DTOs being defined an explanation should be given explaining why this is not applicable in this case.

This section is optional and could also be provided in a separate RFC.

7 Javadoc

Please include Javadoc of any new APIs here, once the design has matured. Instructions on how to export Javadoc for inclusion in the RFC can be found here: https://www.osgi.org/members/RFC/Javadoc

8 Considered Alternatives

For posterity, record the design alternatives that were considered but rejected along with the reason for rejection. This is especially important for external/earlier solutions that were deemed not applicable.

9 Security Considerations

Description of all known vulnerabilities this may either introduce or address as well as scenarios of how the weaknesses could be circumvented.

10 Document Support

10.1 References

[1]. Bradner, S., Key words for use in RFCs to Indicate Requirement Levels, RFC2119, March 1997.

[2]. Software Requirements & Specifications. Michael Jackson. ISBN 0-201-87712-0

Add references simply by adding new items. You can then cross-refer to them by chosing <Insert><Cross Reference><Numbered Item> and then selecting the paragraph. ***STATIC REFERENCES (I.E. BODGED) ARE NOT ACCEPTABLE, SOMEONE WILL HAVE TO UPDATE THEM LATER, SO DO IT PROPERLY NOW.***

10.2 Author's Address

Name	David Bosschaert
Company	Adobe Systems
Address	
Voice	
e-mail	bosschae@adobe.com

Name	Peter Kriens
Company	aQute
Address	
Voice	
e-mail	peter.kriens@aqute.biz

Name	Todor Boev
Company	Software AG
Address	
Voice	
e-mail	Todor.Boev@softwareag.com

Name	
Company	
Address	
Voice	
e-mail	

10.3 Acronyms and Abbreviations

10.4 End of Document