RFP 119 - Fine-grained APIs and Versioned PIDs for the Configuration Admin Service

Draft

9 Pages

Abstract

In today's OSGi Compendium specification, the API available to the Configuration Admin's ManagedService is coarse-grained and doesn't provide any delta information. Many of today's applications have a large set of configuration variables and would benefit from more fine-grained callbacks with additional metadata regarding what has changed.

Additionally, configuration PIDs are currently unversioned entities in the Configuration Admin Service, causing problems when multiple versions of the same bundle are installed simultaneously. This RFP describes these scenarios in more detail and lists requirements that need to be satisfied by a solution to these problems.

0 Document Information

0.1 Table of Contents

0.2 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 6.1.

```
Source code is shown in this typeface.
```

0.3 Revision History

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

Revision	Date	Comments
Initial	17 January 2009	David Bosschaert (Progress), David Grigglestone (Progress), initial version.
0.2	23 Febuary 2009	David Bosschaert (Progress), David Grigglestone (Progress) enhanced to include versioning of config PIDs.
0.3	10 March 2009	David Bosschaert (Progress), changes following feedback from the Austin, TX F2F

1 Introduction

When using the Configuration Admin Service in larger enterprise applications certain shortcomings become apparent. Some of these shortcomings can be worked around, but doing so will cause duplicated code in bundles that use the Configuration Admin. Other shortcomings are more fundamental and prevent the Configuration Admin from being used in ways desirable for the enterprise application. This RFP describes the use cases concerned and the requirements a solution to this problem needs to satisfy.

2 Application Domain

Configuration Management is a comprehensive part of many of today's enterprise applications. Often these applications use a sophisticated configuration management system which is used to enter, record and distribute configuration information.

The OSGi Configuration Admin specification is an appealing technology for such applications since its one of the few standards in the configuration domain.

This document describes proposed enhancements to the Configuration Admin specification to better align its capabilities with enterprise use-cases.

2.1 Terminology + Abbreviations

Configuration PID: The Configuration Admin Service requires that every configuration set (actually a map of name-value pairs) be assigned a unique, persistent, ID. Typically Service PID's have been used for this purpose, but other PID's are also possible. See also RFC 144.

Configuration Target: The target (bundle or service) that will receive the configuration information. For services, there are currently two types of targets: ManagedServiceFactory or ManagedService objects. The realization of the requirements in this document may introduce new configuration targets.

3 Problem Description

3.1 Provide Delta information in Configuration Target callbacks

When using the Configuration Admin Service to configure the components of large systems, it becomes apparent that certain APIs could be improved to better provide the consumers of configuration information with what has actually changed in the configuration.

The current Configuration Admin specification always calls consumers of configuration information (implementors of the ManagedService or ManagedServiceFactory) with the full map of configuration information. Even thought only a subset of the configuration information may have changed. This requires every implementor of the ManagedService to keep a copy of the previous map and implement logic to compute the deltas between the current callback and the previous callback. It would benefit the system if this functionality was provided by the Configuration Admin Service.

3.2 Updates concerning multiple Configuration PIDs

Large systems can have hundreds or sometimes thousands of configuration variables. Typically such systems are modularized in some way (e.g. using OSGi bundles), where some configuration information is shared across modules and other configuration is private to the module.

The Configuration Admin Service allows a Managed Service to be a consumer of multiple configuration sets (configuration PIDs), but it does not provide an API capable of notifying the ManagedService of changes to multiple configuration sets in a single callback. This would be necessary if there is a dependency of configuration values across the configuration sets.

3.3 Multiple versions and multiple instances of Configuration data

In today's Configuration Admin specification, only one instance of Configuration Data exists in the system for a given PID. This can cause issues when multiple versions of the same bundle are installed in the system, especially if the configuration data required by these versions of the bundle is not compatible, or if the configuration values to be provided to the various instances of the bundle need to be different.

4 Use Cases

4.1 Being able to consume fine-grained configuration changes

An Enterprise application has many configuration settings. Certain settings are very fine grained and can be changed at runtime, such as a buffer size, or a logging level.

Today, when such a configuration change is done, the ManagedService receives an updated(Dictionary d) callback with the whole map of configuration variables. Currently, the onus is on ManagedService implementation to figure out what configuration variable has changed and react appropriately. It would make the ManagedService implementation much simpler if the callback provided information about the delta's between this one and the previous callback.

More specifically, providing deltas would save every ManagedService implementation from keeping a map of all the values and every time a callback is made recomputing the delta, while this information is potentially already available to the Configuration Admin implementation. Additionally by requiring the Configuration Admin implementation to manage the deltas, efficiencies will naturally occur when more than one service consumes the same configuration settings.

4.2 Reverting back to defaults for a configuration setting

An application has a number of configuration settings. Some of these settings have default values, for example the memory buffer size used by a certain component has a default provided by the application. Under certain circumstances this default needs to be changed temporarily, especially to facilitate diagnosing of the operation of the component. However, after some time reverting back to the default value is best for the component, as using the default value permits the enabling of certain optimizations.

The callback provided today by the Configuration Admin Service to a ManagedService simply provides the map of all the values. It would benefit the application to receive a more fine-grained callback that indicates what happened to the configuration variables, more specifically it should be able to tell the ManagedService that a particular variable has been removed to allow the application to revert back to the default setting for this variable.

4.3 Update clarity when consuming many configuration PIDs

A ManagedService can be a consumer of multiple configuration PIDs (see bug 1023 https://www.osgi.org/members/bugzilla/show_bug.cgi?id=1023), however the updated(Dictionary d) callback doesn't provide information around which configuration object (PID) triggered the callback. The ManagedServiceFactory does provide an updated(String pid, Dictionary properties), but the ManagedService doesn't.

Providing the configuration PID on the callback would help the ManagedService processing the configuration information in the case it is a consumer to multiple configuration PID.

4.4 Updating multiple configuration PIDs atomically

In some cases configuration variables are related. For example, when making changes to the settings of a thread pool, the high watermark and the low watermark typically need to be set at the same time.

This would even sometimes be required across multiple configuration PIDs. A component may have shared configuration, such as the location to store its data files, plus private configuration such as the name of its own data file. This configuration information needs to be provided to the component in a single callback to prevent the component from creating a data file with the wrong name (e.g. the default name) in the storage location.

4.5 Bundle Upgrade and Roll-back

An OSGi system contains bundle A. The installed version of bundle A is 1.0. At a certain point in time a new version 2.0 of bundle A is released and this new version requires new configuration data which is not compatible with the configuration data of bundle A version 1.0. As usual with OSGi bundle updates, the new version 2.0 bundle is installed and started first, alongside version 1.0. Once version 2.0 is properly installed and configured bundle A version 1.0 will be uninstalled.

After a while the users of the system see a regression in functionality and this is tracked down to the new bundle A version 2.0. The system administrator decides to revert the upgrade and reinstall version 1.0 of bundle A, and then uninstall version 2.0.

In today's OSGi containers the above scenario poses a problem if bundle A consumes configuration from the OSGi Configuration Admin Service using a stable PID without any version information, as the same configuration instance will have to be shared by both version 1.0 and 2.0 of the bundle. Adding version information to the PID will help bundle upgrade and roll-back scenarios like this one as it will be possible to create two distinct configuration objects, one for version 1.0 of bundle A and another one for version 2.0 of bundle A.

4.6 Multiple bundles sharing the same configuration object

An OSGi container hosts both bundle A and bundle B who share a configuration object with PID x.y.z. They achieve this by specifying the same configuration PID when registering a ManagedService. RFC 144 has made this possible.

At a certain point in time bundle B needs to be upgraded to bundle B'. This upgrade requires a change in the configuration data which makes it incompatible with the old version of the data.

Given the fact that both bundle A and B' use the same configuration PID, it is impossible to satisfy both of their configuration requirements. Versioning the PIDs would help as it would allow different instances of configuration data for a given PID to coexist.

4.7 Multiple versions of the same bundle co-existing

An OSGi container in an Enterprise system has a number of bundles installed among which bundle A and bundle B. Bundle A has a dependency on bundle C version 1. Bundle B has a dependency on bundle C version 2. So both version 1 and version 2 of bundle C are installed in the OSGi container.

Bundle C uses the Configuration Admin service to obtain its configuration. However the configuration data has changed significantly for bundle C between version 1 and version 2. This poses a problem since bundle C uses the same configuration PID for both versions 1 and 2. Today this will cause the same configuration object to be served to both instances of bundle C, causing one of the instances to see the wrong version of the data. Adding version information to the PID and holding an Configuration Data object per PID+version combination in the Configuration Admin Service would solve this situation.

5 Requirements

FGCA00. The solution depends on a feature of the Configuration Admin Service introduced in RFC 144: the capability of multiple bundles being a able to consume a single Configuration PID.

FGCA01. The solution MUST provide delta information when calling back with configuration updates.

FGCA02. The solution MUST provide the configuration PID on all newly defined callbacks.

FGCA03. The solution MUST provide the capability of combining multiple configuration changes in a single callback to the ManagedService.

FGCA04. The solution MUST provide an API that allows a single update callback for multiple configuration PIDs on a particular Configuration Target.

FGCA05. The solution SHOULD be applied to both the factory configurations as well as normal configurations.

FGCA06. It MUST be possible to have multiple versions of configuration data exist concurrently for a given PID.

FGCA07. It SHOULD be possible for a Configuration Target to specify a version range for a suitable configuration object, given a PID.

FGCA08. It MUST be possible for a Configuration Target to specify the exact version of a suitable configuration object, given a PID.

FGCA09. It MUST be possible for two versions of the same bundle to use separate configuration objects for a given PID if the configuration PID versions used are different.

FGCA10. It MUST be possible for two versions of the same bundle to share the same configuration object for a given PID if the configuration PID versions are the same.

FGCA11. The solution MUST be backwards compatible. This implies that the configuration consumer must also be able to use the system without specifying the PID version.

FGCA12. The existing Configuration Admin interfaces SHOULD be updated to support versioned PIDs.

FGCA13. The Configuration Admin interfaces SHOULD be enhanced to support setting configuration deltas. This includes atomically setting deltas across configuration PIDs.

FGCA014. The solution MUST provide the configuration version on all newly defined callbacks.

FGCA015. The solution MUST define what configuration, or configurations, will be provided to a Configuration Target in the event of multiple versions matching a version range.

6 Document Support

6.1 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

6.2 Author's Address

Name	David Bosschaert
Company	Progress Software
Address	The IONA Building, Shelbourne Road, Ballsbridge, Dublin 4, Ireland
Voice	+353 1 637 2000
e-mail	dbosscha@progress.com

Name	David Grigglestone
Company	Progress Software
Address	14 Oak Park Drive, Bedford, MA 01730, USA
Voice	+1 781-280-4000
e-mail	davidg@progress.com

6.3 End of Document