RFC 49 Supervisioning API

Confidential, Draft
RFC 49

6 Pages

Abstract

There is currently no way for the OSGi framework or management bundles to detect if a bundle is functioning as intended. To detect a bundle malfunction, detailed knowledge about what a bundle is supposed to do is needed. It is only the bundle itself that has such knowledge. This API allows a bundle to expose how well it is functioning to interested parties.

0 Document Information

0.1 Table of Contents

0.2 Status

This document specifies a bundle super visioning API for the Open Services Gateway Initiative, and requests discussion and suggestions for improvements. Distribution of this document is unlimited within OSGi.

0.3 Acknowledgement

This document is based upon the contributions of CPEG members.

0.4 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 [1].

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

0.5 Revision History

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

Revision	Date	Comments
Initial	Apr 15 2002	Initial draft. Gunnar Ekolin, Gatespace, ekolin@gatespace.com Jan Sparud, Gatespace, sparud@gatespace.com

1 Introduction

There is currently no way for the OSGi framework or management bundles to detect if a bundle is functioning as intended. To detect a bundle malfunction, detailed knowledge about what a bundle is supposed to do is needed. It is only the bundle itself that has such knowledge. This API allows a bundle to expose how well it is functioning to interested parties.

2 Motivation and Rationale

A bundle may be in one of several states as defined by OSGi. However, there is only one state (`Bundle.ACTIVE`) describing a running bundle. There is no way for a bundle to announce that it is malfunctioning. Because of this, there is no way for a management bundle to detect that the bundle is malfunctioning. Only the bundle itself has access to its state, and can make judgments whether it is running as it is supposed to.

This proposal addresses parts of OSGi RFP0034 [2].

3 Technical Discussion

The proposed API allows a bundle to register a service that exposes the bundle's own view about its internal state. A management bundle may periodically probe the supervised bundles by using this API. It is then up to the management bundle to take action, if a bundle reports that it is malfunctioning.

Bundles may malfunction in many ways. It is impossible, or at least hard, to define a semantics of the possible failure modes of bundles. For this reason, we have decided to have only two levels of soundness. A bundle may report:

- That it is working as intended.
- That is not fully functional. In this case it may give a description of the perceived problem.

3.1 org.osgi.service.supervision Interface Supervise

public interface **Supervise**

Interface to register as a service by bundles that want to make their internal state available to management bundles.

A management bundle will call the `getState()` method defined in this interface and if the return value is `null` the bundle exporting the service will be considered fully functional. If the return value is a non-`null` String the service has limited functionality. The string describes the problem.

All Page Within This Box

Method Summary

`Java.lang.String`	**`getState`**`()` Bundle state polling method.

Method Detail

3.1.1 getState

```
public java.lang.String getState()
```

Bundle state polling method. Implementations must return `null` if the bundle exporting this service is fully functional. All other results, including thrown exceptions, are interpreted as if the bundle is malfunctioning.

Returns:

`null` if bundle exporting this service is fully functional, otherwise a message describing the limits in the functionality.

4 Security Considerations

Bundles that do not want to make their internal state available to any other bundle, may require that callers have AdminPermission.

5 Document Support

5.1 References

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

[2]. Ekolin, Sparud, OSGi RFP 0034 Fault Management Foundation

5.2 Author's Address

Name	Gunnar Ekolin
Company	Gatespace
Address	Stora Badhusgatan 18-20; SE41121 Göteborg; Sweden
Voice	
e-mail	ekolin@gatespace.com

Name	Jan Sparud
Company	Gatespace
Address	Stora Badhusgatan 18-20; SE41121 Göteborg; Sweden
Voice	
e-mail	sparud@gatespace.com

5.3 Acronyms and Abbreviations

5.4 End of Document