Service Scoping (v2.0)

Rob Harrop

November 19, 2008

This document discusses the specification, design and implementation of service scoping as present in dm Server version 2.0.

Contents

1 Overview

Service scoping is an extension to standard OSGi service binding that introduces preferential selection and visibility restriction.

A scope defines a grouping for services. Scopes are split in two kinds: application and global. There is a single global scope and zero or more application scopes.

Within a given application scope, all published services are visible only in that scope. This is visibility restriction.

When a component in an application scope performs a service lookup, any matching services from within the scope are preferred over matching services in the global scope.

2 Services and Modules

Services are opaque, unique objects

$[\mathit{Service}]$

$\mathit{Filter} == \mathbb{P}\,\mathit{Service}$

$$\begin{array}{|l}
\hline \mathit{Module} \\
\mathit{exp} : \mathbb{P}\,\mathit{Service} \\
\mathit{imp} : \mathbb{P}\,\mathit{Service} \\
\mathit{global} : \mathbb{P}\,\mathit{Service} \\
\hline
\mathit{exp} \subseteq \mathit{global} \\
\hline
\end{array}$$

$$\begin{array}{|l}
\hline \mathit{StandardPublish} \\
\Delta \mathit{Module} \\
s? : \mathit{Service} \\
\hline
\mathit{imp}' = \mathit{imp} \\
\mathit{exp}' = \mathit{exp} \cup \{s?\} \\
\mathit{global}' = \mathit{global} \cup \{s?\} \\
\hline
\end{array}$$

$$\begin{array}{|l}
\hline \mathit{StandardBind} \\
\Delta \mathit{Module} \\
f? : \mathit{Filter} \\
\hline
f? \cap \mathit{global} \neq \emptyset \\
\mathit{global}' = \mathit{global} \\
\mathit{exp}' = \mathit{exp} \\
\mathit{imp}' = \mathit{imp} \cup (f? \cap \mathit{global}) \\
\hline
\end{array}$$

$$\begin{array}{|l}
\hline \mathit{ScopedModule} \\
\mathit{Module} \\
\mathit{scopedExp} : \mathbb{P}\,\mathit{Service} \\
\hline
\mathit{scopedExp} \cap \mathit{exp} = \emptyset \\
\mathit{scopedExp} \cap \mathit{global} = \emptyset \\
\mathit{imp} \subseteq \mathit{scopedExp} \cup \mathit{global} \\
\hline
\end{array}$$

$$\begin{array}{|l}
\hline \mathit{GlobalPublish} \\
\Delta \mathit{ScopedModule} \\
\mathit{StandardPublish} \\
\hline
s? \notin \mathit{scopedExp} \\
\mathit{scopedExp}' = \mathit{scopedExp} \\
\hline
\end{array}$$

ScopedPublish

$\Delta ScopedModule$
$\Xi Module$
$s? : Service$

$s? \notin global$
$scopedExp' = scopedExp \cup \{s?\}$

GlobalBind

$\Delta ScopedModule$
$StandardBind$

$f? \cap scopedExp = \emptyset$
$scopedExp' = scopedExp$

ScopedBind

$\Delta ScopedModule$
$f? : Filter$

$f? \cap scopedExp \neq \emptyset$
$global' = global$
$exp' = exp$
$scopedExp' = scopedExp$
$imp' = imp \cup (f? \cap scopedExp)$

3 Design Overview

When a user deploys a PAR artefact, all modules in that PAR are grouped into the same scope. This scope is called the **application scope**. Each PAR defines a discrete application and therefore a discrete application scope.

Application scopes are named - the name is derived from the `Application-SymbolicName` and `Application-Version` defined in the PAR file.

Modules that are deployed outside of a PAR file are said to reside in the global scope.

3.1 Publication

Application-scoped modules can choose to publish services in their own scope or in the global scope. Globally-scoped modules can only publish services into the global scope.

A scoped module can choose which scope a service is to be published into the service property `com.springsource.service.scope` to the value `global` for global scope or `app` for application scope.

Services published using Spring DM from an application-scoped bundle are automatically application scoped. Programmatic service publication from an application-scoped bundle defaults to using the global scope.

Globally-scoped modules can never force a service to be published inside an application scope.

3.2 Binding

Globally scoped modules can only bind to services that are published in the global scope. No application scoped services are visible to globally scoped modules.

When binding to services, application-scoped modules will always prefer matching services published in their own scope over matching services in the global scope. If no matching services are found in the application's scope, then binding can proceed as normal against the global scope.

Application-scoped modules can never bind to services published in another application's scope.

4 Implementation

The chosen implementation uses the service registry hooks to restrict service visibility on service lookup and an publication of listener events.

4.1 Service Lookup

Restricting visibility during service lookup is done using a `FindHook`. The `FindHook` gives the implementor the ability to restrict the `ServiceReferences` that would normal be returned during a lookup. This is done by iterating over a `Collection` of `ServiceReferences` that match the user's filter and removing references as necessary.

To determine whether a given `ServiceReference` should be removed we need to determine the scope that the `ServiceReference` is published under and the scope that the lookup is being performed in.

The scope that a service is published under can be determined using the following rules:

1. If the service was published by a globally-scoped module, then the service is globally-scoped, otherwise;
2. If the service has the `com.springsource.service.scope` property set then the value of that property determines the scope, otherwise;
3. If the service has the `org.springframework.osgi.bean.name` property set, indicating that it was published by Spring DM, then the service is application-scoped, otherwise;
4. The service is globally-scoped

The lookup scope is calculated using the following rules:

1. If the consumer bundle is globally-scoped, then the lookup is globally-scoped, otherwise;
2. If the user-supplied filter matches a service exported by any bundle in the scope as the consumer then the lookup is application-scoped, otherwise;
3. The lookup is globally-scoped

`ServiceReferences` are removed from the `Collection` supplied by the `FindHook` if their publication scope does not match the lookup scope.

4.2 Listeners

Restricting visibility for listeners is very simple. If the publication scope for a service is global then all listeners can see events for that service. If the publication scope for a service is application, then only listeners registered by bundles in the same scope can see events for the service.

4.3 Matching Against Exported Services

discuss service model

5 Implementation Alternatives

The original service scoping implementation used XML rewriting to add service properties and filters to Spring DM ¡service¿ and ¡reference¿ tags. These properties and filters were used to constrain the visibility of services to an application to ensure that application services were selected over global services.

This approach has the following drawbacks:

1. No meaningful model can be defined for programmatic access
2. Rewriting XML necessitated taking copies of user artefacts. This makes redeploying on user change overly difficult
3. The scoping is easy to break if the right properties and filters are known