

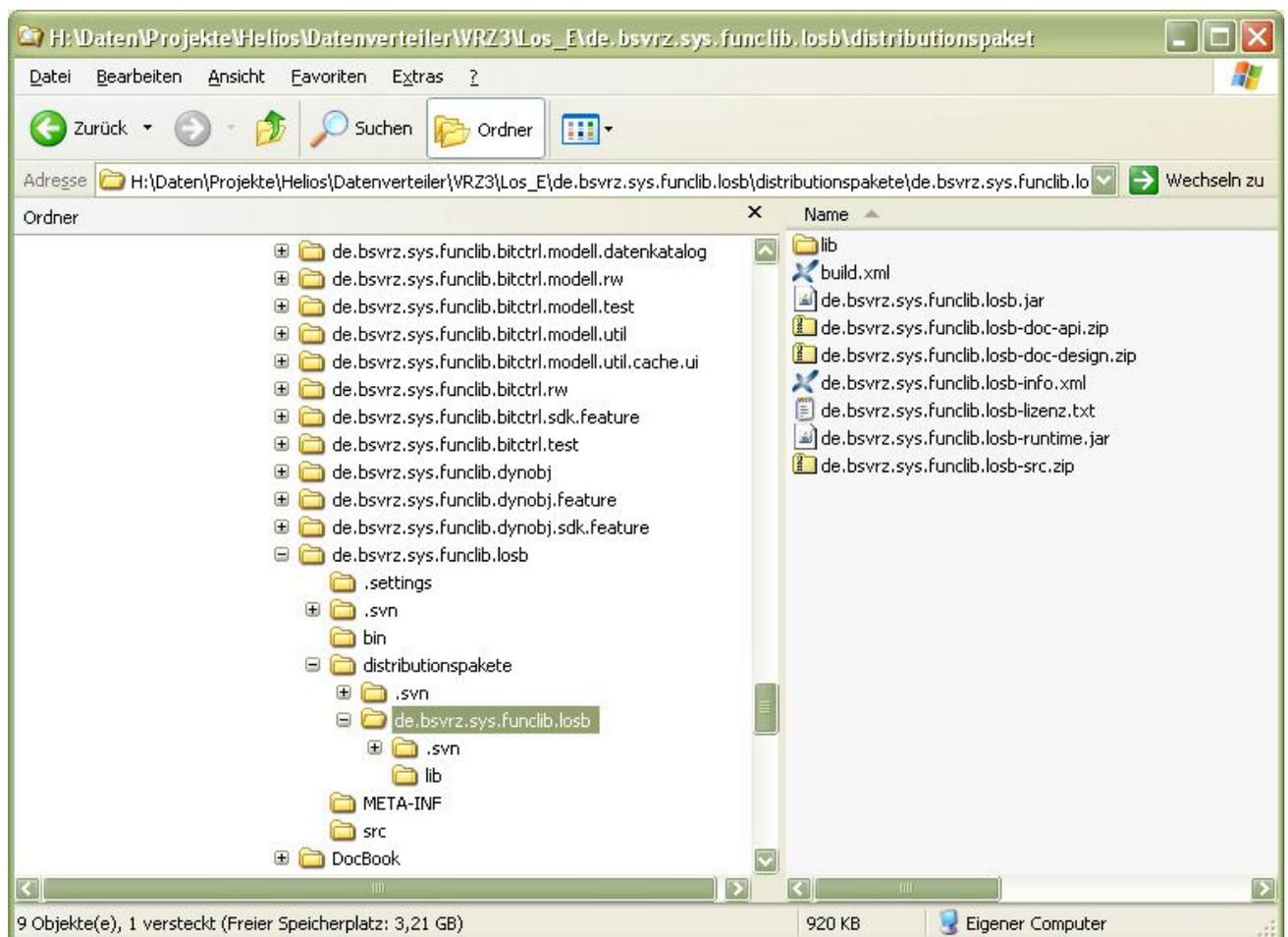
Bereitstellung von Bibliotheken für das BS-VRZ-Rahmenwerk

Damit die notwendigen Software-Bibliotheken in den verschiedenen SWE des BS-VRZ-Rahmenwerks verwendet werden können, müssen die Bibliotheken vom jeweiligen Hersteller parallel als Plug-ins für das Rahmenwerk zur Verfügung gestellt werden. Genau genommen, müssen die Plug-ins als Feature, das sich über eine Updateseite installieren und pflegen lässt, zur Verfügung gestellt werden.

Der Aufwand aus Sicht der Softwareentwicklung hält sich dabei in Grenzen, da üblicherweise ohnehin nur in relativ großen Abständen neue Versionen zur Verfügung gestellt werden und die "Verpackung" der Software nicht wirklich berührt wird.

Im Folgenden ist kurz prototypisch die Bereitstellung der SWE "de.bsvrz.sys.funclib.losb" dargestellt.

Die Struktur des Plug-ins ist unten in dem Abbild des Windows-Explorers dargestellt:



Die Verzeichnisse ".settings" und ".svn" können vernachlässigt werden. Settings enthält Informationen zur Eclipse-IDE-Konfiguration für das Plugin. Die SVN-Verzeichnisse stammen vom Subversion-Repository.

Die Verzeichnisse "bin" und "src" sind leer und nur entstanden, weil sie standardmäßig beim Anlegen eines Plug-ins in der Eclipse-IDE angelegt werden.

Das Verzeichnis "distributionspakete" enthält die SWE in der Form, wie sie auch im Datenverteiler (ohne Rahmenwerk) verwendet wird.

Lediglich die Manifestdatei im Verzeichnis META-INF muss entsprechend angepasst werden. Das kann am einfachsten über die Eclipse-IDE erfolgen. Nachfolgend sind die vorzunehmenden Einstellungen beschrieben:

Overview

The screenshot shows the Eclipse IDE's 'Overview' page for a plug-in. The browser tab is 'de.bsvrz.sys.funclib.losb'. The page title is 'Overview'. The main content is organized into several sections:

- General Information:** This section describes general information about the plug-in. It includes fields for ID (de.bsvrz.sys.funclib.losb), Version (1.0.0.qualifier), Name (Los B - Bibliothek), Provider (BitCtrl Systems GmbH), Platform Filter, and Activator. There are checkboxes for 'Activate this plug-in when one of its classes is loaded' and 'This plug-in is a singleton'.
- Execution Environments:** This section specifies the minimum execution environments required to run the plug-in. It shows a list of environments with 'JavaSE-1.6' selected. There are buttons for 'Add...', 'Remove', 'Up', and 'Down'. Below the list are links for 'Configure JRE associations...' and 'Update the classpath settings'.
- Plug-in Content:** This section explains the content of the plug-in. It states that the content is made up of two sections: 'Dependencies' (lists all the plug-ins required on this plug-in's classpath to compile and run) and 'Runtime' (lists the libraries that make up this plug-in's runtime).
- Extension / Extension Point Content:** This section describes how the plug-in may define extensions and extension points. It includes links for 'Extensions' (declares contributions this plug-in makes to the platform) and 'Extension Points' (declares new function points this plug-in adds to the platform).
- Testing:** This section provides instructions on how to test the plug-in by launching a separate Eclipse application. It includes links for 'Launch a RAP Application', 'Launch an Eclipse application', 'Launch a RAP Application in Debug mode', and 'Launch an Eclipse application in Debug mode'.
- Exporting:** This section describes how to package and export the plug-in. It includes a list of steps: 1. Organize the plug-in using the 'Organize Manifests Wizard', 2. Externalize the strings within the plug-in using the 'Externalize Strings Wizard', 3. Specify what needs to be packaged in the deployable plug-in on the 'Build Configuration' page, and 4. Export the plug-in in a format suitable for deployment using the 'Export Wizard'.

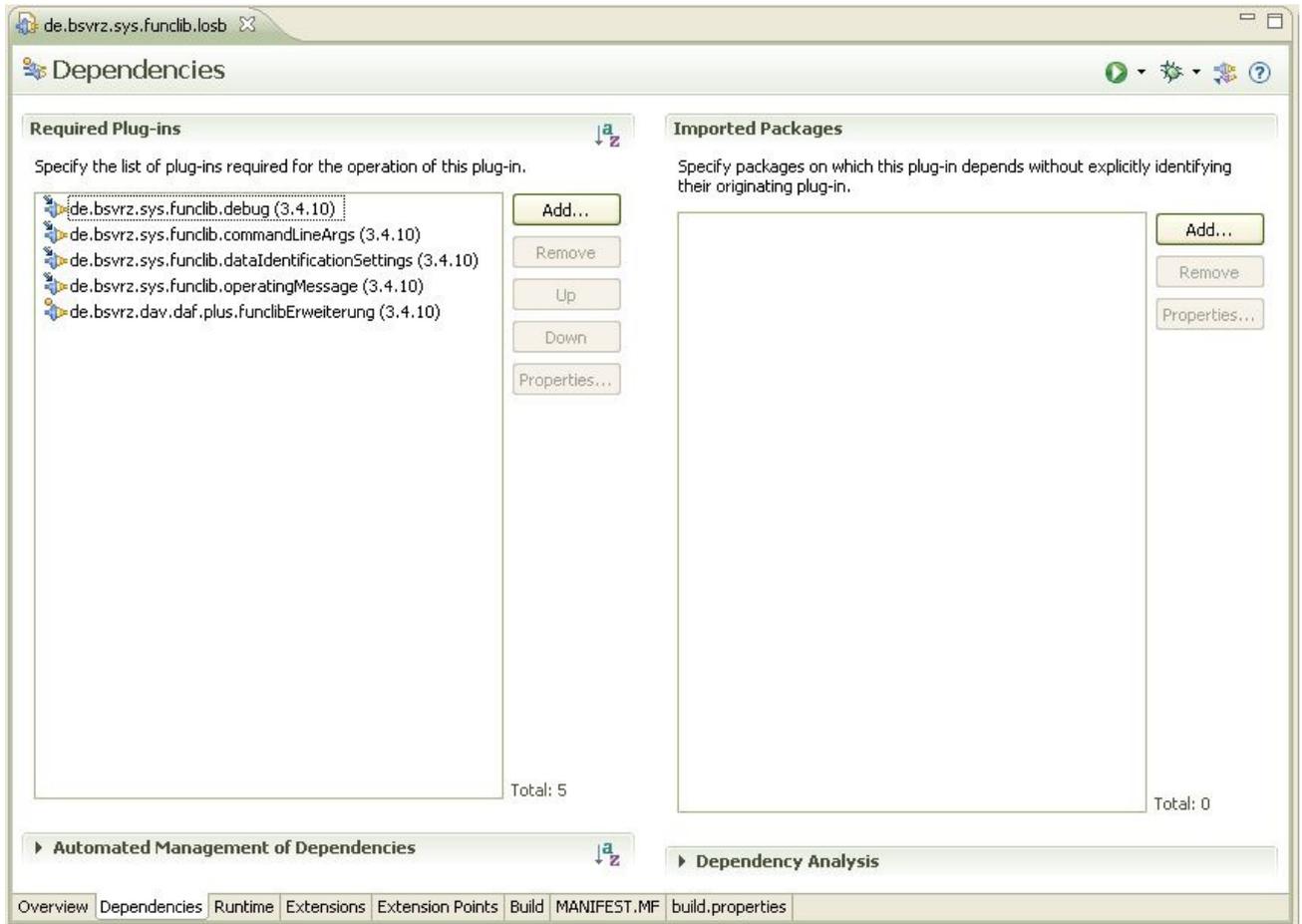
At the bottom of the page, there is a navigation bar with tabs for 'Overview', 'Dependencies', 'Runtime', 'Extensions', 'Extension Points', 'Build', 'MANIFEST.MF', and 'build.properties'.

Die Seite enthält allgemeine Informationen für das Plug-in.

Wichtig sind die **Plug-in-ID** über die andere Plug-ins ihre Abhängigkeiten zu diesem definieren können.

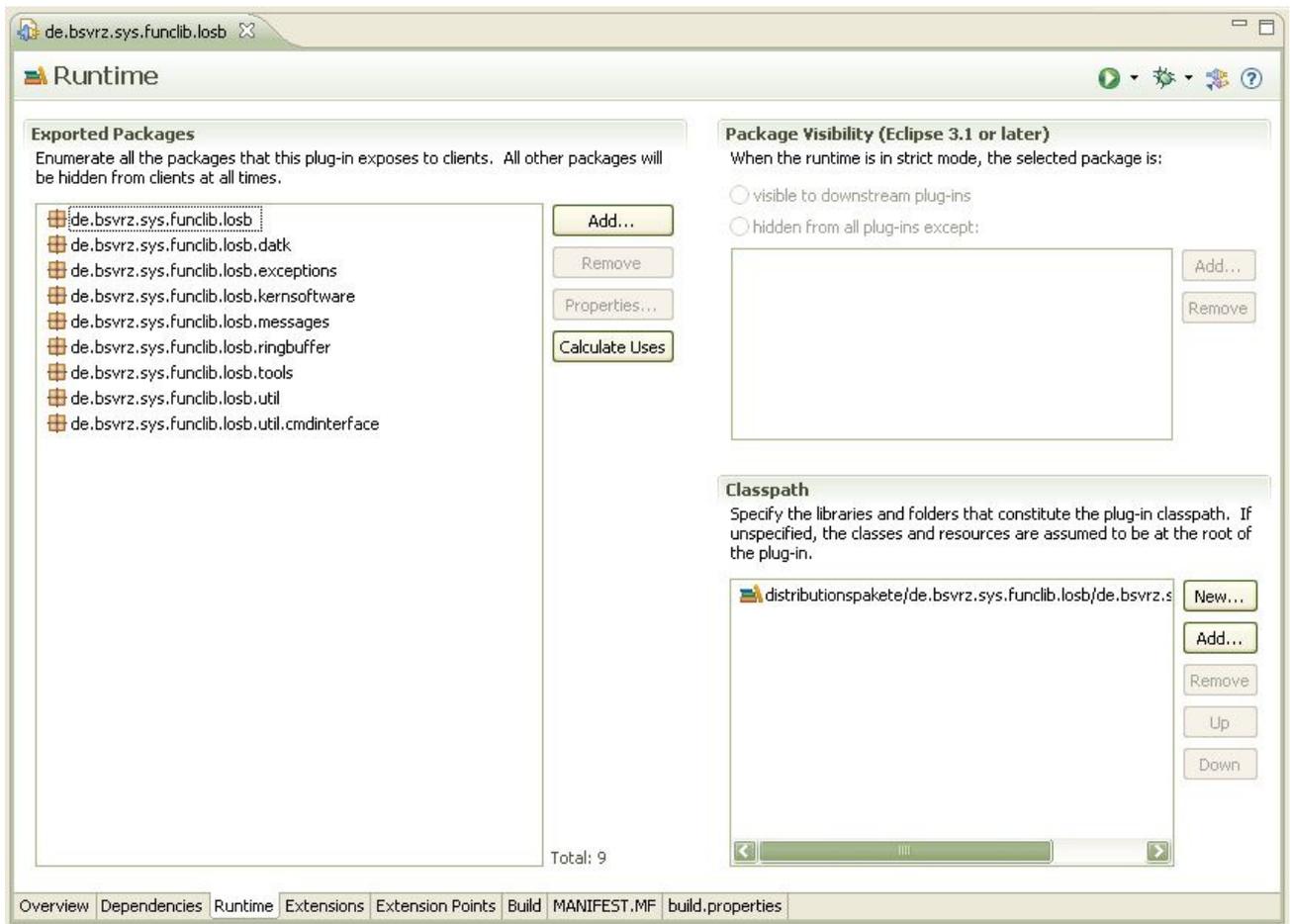
Zu beachten ist die korrekte Vergabe der Versionsnummer, da diese vom Update-Mechanismus berücksichtigt wird. Es werden nur Plug-ins aktualisiert, die eine höhere Versionsnummer haben. Die Erweiterung "qualifier" in der Versionsbezeichnung wird beim Erstellen des Plug-ins als Paket durch das aktuelle Datum/Zeit ersetzt, d. h. es muss nicht für laufende Änderungen händisch die Versionsnummer erhöht werden. Im Releasefall sollte dies aber auf jeden Fall passieren.

Dependencies



enthält alle Plug-ins, von denen dieses selbst abhängig ist.

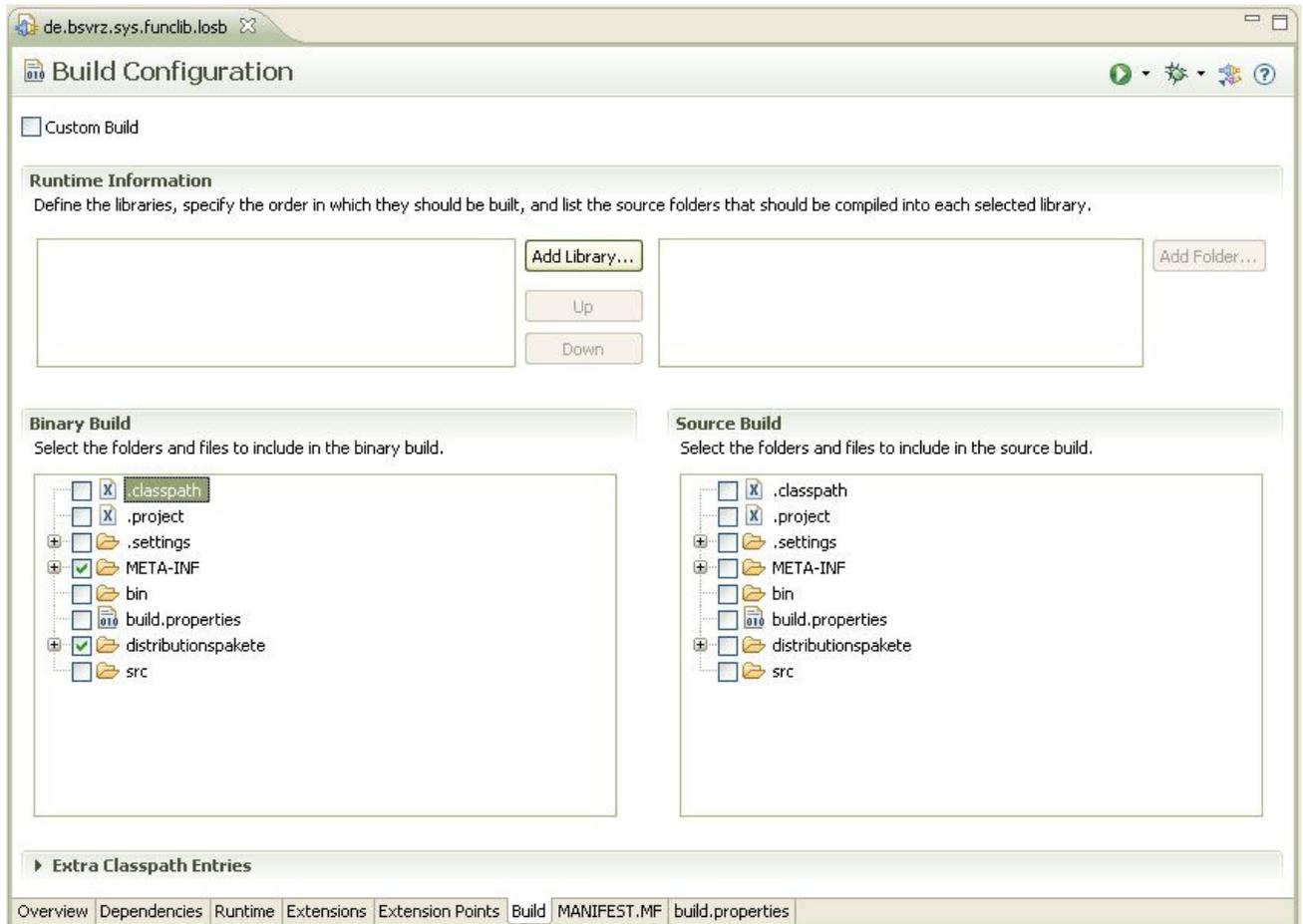
Runtime



listet alle Packages auf, die vom Plug-in bereitgestellt werden, d. h. die von anderen Plug-ins verwendet werden können.

Unter "Classpath" wird der Pfad zur Bibliothek (Jar-file) angegeben.

Build Configuration



definiert, welche Teile des Projekts in das zu veröffentlichende Plug-in verpackt werden.

MANIFEST.MF

```
1 Manifest-Version: 1.0
2 Bundle-ManifestVersion: 2
3 Bundle-Name: Los B - Bibliothek
4 Bundle-SymbolicName: de.bsvrz.sys.funclib.losb
5 Bundle-Version: 1.0.0.qualifier
6 Bundle-Vendor: BitCtrl Systems GmbH
7 Bundle-RequiredExecutionEnvironment: JavaSE-1.6
8 Eclipse-BuddyPolicy: registered
9 Export-Package: de.bsvrz.sys.funclib.losb,
10 de.bsvrz.sys.funclib.losb.datk,
11 de.bsvrz.sys.funclib.losb.exceptions,
12 de.bsvrz.sys.funclib.losb.kernsoftware,
13 de.bsvrz.sys.funclib.losb.messages,
14 de.bsvrz.sys.funclib.losb.ringbuffer,
15 de.bsvrz.sys.funclib.losb.tools,
16 de.bsvrz.sys.funclib.losb.util,
17 de.bsvrz.sys.funclib.losb.util.cmdinterface
18 Bundle-ClassPath: distributionspakete/de.bsvrz.sys.funclib.losb/de.bsvrz.sys.funclib.losb.jar
19 Require-Bundle: de.bsvrz.sys.funclib.debug;bundle-version="3.4.10",
20 de.bsvrz.sys.funclib.commandLineArgs;bundle-version="3.4.10",
21 de.bsvrz.sys.funclib.dataIdentificationSettings;bundle-version="3.4.10",
22 de.bsvrz.sys.funclib.operatingMessage;bundle-version="3.4.10",
23 de.bsvrz.dav.daf.plus.funclibErweiterung;bundle-version="3.4.10"
24
```

bietet die Möglichkeit, die Manifestdatei direkt zu editieren (was mit den vorher beschriebenen Dialogen interaktiv erfolgt).

Ein Eintrag, der nicht über die Klickoberfläche erfolgen kann und deshalb händisch zu ergänzen ist, ist von besonderer Bedeutung, wenn innerhalb des Plug-ins die Reflection-API gearbeitet wird.

Eclipse-BuddyPolicy: registered

Innerhalb des Eclipse/OSGI-Framework sind Abhängigkeiten zwischen den Plug-ins definiert festgelegt, d. h. es kann keine Instanz einer Klasse per Reflection erzeugt werden, die in einem "fremden" Paket liegt. Um das zu ermöglichen, muss das Plug-in erlauben, dass zur Laufzeit zusätzliche Ressourcen (Buddys) für den Classloader registriert werden können. Genau das erlaubt dieser Eintrag.

Siehe auch: http://wiki.eclipse.org/index.php/Context_Class_Loader_Enhancements

Zur Installation der Plug-ins sollten diese in einem oder mehreren Features zusammengepackt werden. Eine kurze Beschreibung, wie das zu tun ist gibt es beispielsweise hier: <http://www.vogella.de/articles/EclipseFeatureProject/article.html>

Aus diesem Feature kann ganz einfach eine Updateseite generiert werden, über die dann die Installation und die Aktualisierung im Rahmenwerk erfolgen kann.

U. Peuker

BitCtrl Systems GmbH
Weißenfels Str. 67
04229 Leipzig, Germany

E-Mail: Uwe.Peuker@bitctrl.de
Internet: www.bitctrl.de, www.bitctrl.com
Tel: +49 341-490670
Fax: +49 341-4906715

© 2011 BitCtrl