

Publishing BIRT to Maven Central

BIRT has a complex release engineering process that has evolved over several decades with effectively no documentation. The complexity compounded by the complete lack of people with historical knowledge about the details. The dependencies were poorly managed and although that has slowly improved, more investment is required. The complexity is further amplified by a plethora of poorly documented ant tasks that produce artifacts whose use case is not well understood and is not documented.

There appears to have been one recent attempt to publish BIRT of Maven central for version 4.9.0 but that looks to be effectively a unstructured dump.

<https://repo1.maven.org/maven2/org/eclipse/birt>

My impression is that the artifacts and metadata are of such poor quality as to be effectively useless.

There also appear to be older attempts, but those too look like dumps:


























































<https://repo1.maven.org/maven2/org/eclipse/birt/runtime/>

The task of publishing BIRT to Maven Central begs the question, which artifacts are important to publish and which are not? E.g., does it make sense to publish to Maven Central bundles that are effectively only useful in a running Eclipse IDE? I think probably not.

One might hope that BIRT's feature definitions offer some guidance, but even those do a poor job of defining what is actually core runtime versus IDE integration with IDE dependencies sprinkled into the mix:

<https://github.com/eclipse-birt/birt/blob/85b3299a63c439963b4766b2fe01b214ac613567/features/org.eclipse.birt.osgi.runtime/feature.xml#L215-L220>

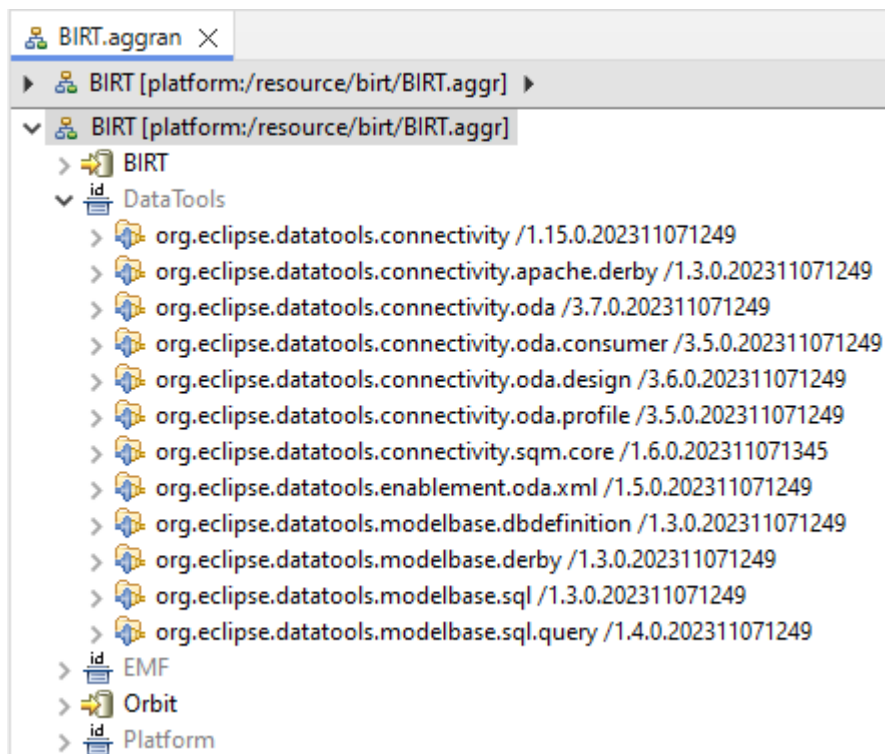
I propose that we publish only those components with no Eclipse IDE dependencies, i.e., this subset.

- >  org.eclipse.birt.chart.device.extension /4.15.0.v202403260939
- >  org.eclipse.birt.chart.device.pdf /4.15.0.v202403260939
- >  org.eclipse.birt.chart.device.svg /4.15.0.v202403260939
- >  org.eclipse.birt.chart.device.swt /4.15.0.v202403260939
- >  org.eclipse.birt.chart.engine /4.15.0.v202403260939
- >  org.eclipse.birt.chart.engine.extension /4.15.0.v202403260939
- >  org.eclipse.birt.chart.reportitem /4.15.0.v202403260939
- >  org.eclipse.birt.core /4.15.0.v202403260939
- >  org.eclipse.birt.core.script.function /4.15.0.v202403260939
- >  org.eclipse.birt.data /4.15.0.v202403260939
- >  org.eclipse.birt.data.aggregation /4.15.0.v202403260939
- >  org.eclipse.birt.data.oda.mongodb /4.15.0.v202403260939
- >  org.eclipse.birt.data.oda.pojo /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.adapter /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.bidi.utils /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.oda.excel /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.oda.hive /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.oda.jdbc /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.oda.jdbc.dbprofile /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.oda.jdbc.dbprofile.sampledb /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.oda.sampledb /4.15.0.v202403260939
- >  org.eclipse.birt.report.data.oda.xml /4.15.0.v202403260939
- >  org.eclipse.birt.report.debug.core /4.15.0.v202403260939
- >  org.eclipse.birt.report.designer.samplerreports /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.dataextraction /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.dataextraction.csv /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.docx /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.excel /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.html /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.odp /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.ods /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.odt /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.pdf /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.postscript /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.ppt /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.pptx /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.config.wpml /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.docx /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.html /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.odp /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.ods /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.odt /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.pdf /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.postscript /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.ppt /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.pptx /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.prototype.excel /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.emitter.wpml /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.fonts /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.odf /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.ooxml /4.15.0.v202403260939
- >  org.eclipse.birt.report.engine.script.javascript /4.15.0.v202403260939
- >  org.eclipse.birt.report.item.crosstab.core /4.15.0.v202403260939
- >  org.eclipse.birt.report.model /4.15.0.v202403260939
- >  org.eclipse.birt.report.model.adapter.oda /4.15.0.v202403260939

We might also consider to omit org.eclipse.birt.chart.device.swt given its SWT dependencies probably makes is not an interesting Maven artifact.

From past experience with publishing the Eclipse Platform and the Eclipse Modeling Framework (EMF) to Maven Central, an essential aspect of producing quality consumable artifacts is publishing POMs with proper dependencies, and properly testing that those dependencies resolve.

A cursory dependency analysis of BIRT's core runtime (including chart), groups the dependencies as follows:




The Platform and the EMF publish BIRT's required dependencies to Maven Central, but DataTools does not. That will need to be addressed; it could be addressed by publishing these to BIRT's coordinate space rather than by asking the DataTools project to publish its own artifacts to Maven Central.

The Orbit dependencies are those available via the Orbit project, specifically as made available by these update sites:

<https://download.eclipse.org/tools/orbit/simrel/orbit-aggregation/>





These newer Orbit dependencies are generally derived from Maven Central artifacts and/or correspond to equivalent artifacts already available at Maven Central, so these are not problematic for the publishing process.

This dependency is problematic because GEF does not publish to Maven Central:

>  org.eclipse.draw2d /3.15.0.202402212051

But a quick analysis suggests this dependency is historical garbage that can be eliminated.

These few old/outdated dependencies definitely need attention:

- >  javax.activation /1.1.0.v201211130549
- >  org.apache.commons.codec /1.14.0.v20221112-0806
- >  org.apache.commons.logging /1.2.0.v20180409-1502
- >  org.apache.derby /10.11.1.1_v201605202053

I believe the first three can be easily addressed by using equivalents already available in Orbit. The Derby dependency is significant problem in that the old version exposes consumers to CVEs. There is a much newer version of Derby available:

<https://repo1.maven.org/maven2/org/apache/derby/derby/10.17.1.0/>

Unfortunately that artifact is effectively garbage in terms of being a well-formed OSGi bundle. The work needed for Orbit to produce properly structured Derby artifacts, with sources, is substantial and is similar to what was needed to bundle Axis 1.x and Ant. Orbit will need to get the artifacts along with sources from here and then massage them for repackaging:

https://db.apache.org/derby/derby_downloads.html

One gotcha with this new version is that it requires Java 21 effectively requires BIRT as a whole to move to Java 21. This slightly older version does not:

<https://repo1.maven.org/maven2/org/apache/derby/derby/10.16.1.1/>

But the latest version fixes a CVE...

The Eclipse Platform publishes SNAPSHOTS to repo.eclipse.org:

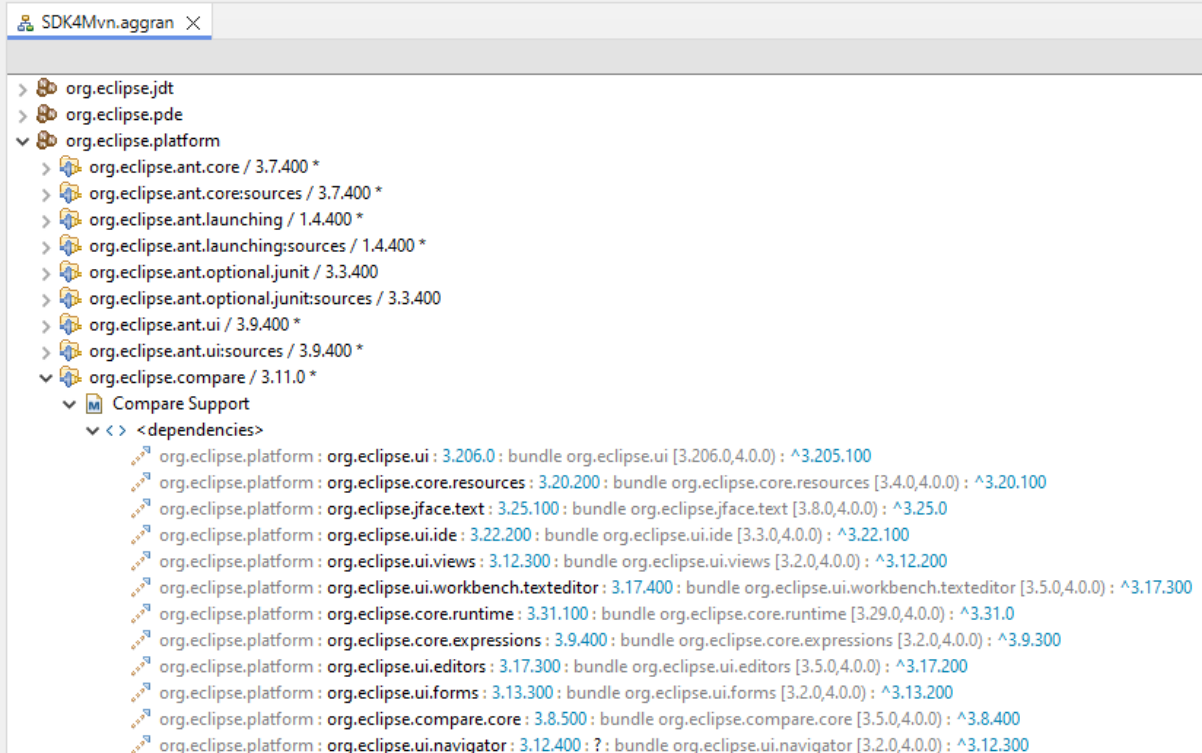
<https://repo.eclipse.org/content/repositories/eclipse-snapshots/org/eclipse/>

It's probably a good idea that BIRT follows a similar process such that the published Maven artifacts are available and can be assessed for integrity **before** the results are ultimately published to Maven Central. It's very problematic that one cannot fix artifacts published to Maven Central, one can only publish new versions with the said fixes, so we really should aim to produce correct results the first time or accept several months delay for publishing a subsequent release correctly.

Proposal

I propose to review BIRT's 3rd party dependencies to ensure they are using only newer Orbit bundles and to provide infrastructure in Orbit for producing OSGi artifacts for newer Derby versions, ensuring that said infrastructure is automated such that consuming and providing newer versions as they become available in the future is a light-weight process. We must be prepared to address security problems by providing and consuming new versions quickly and easily.

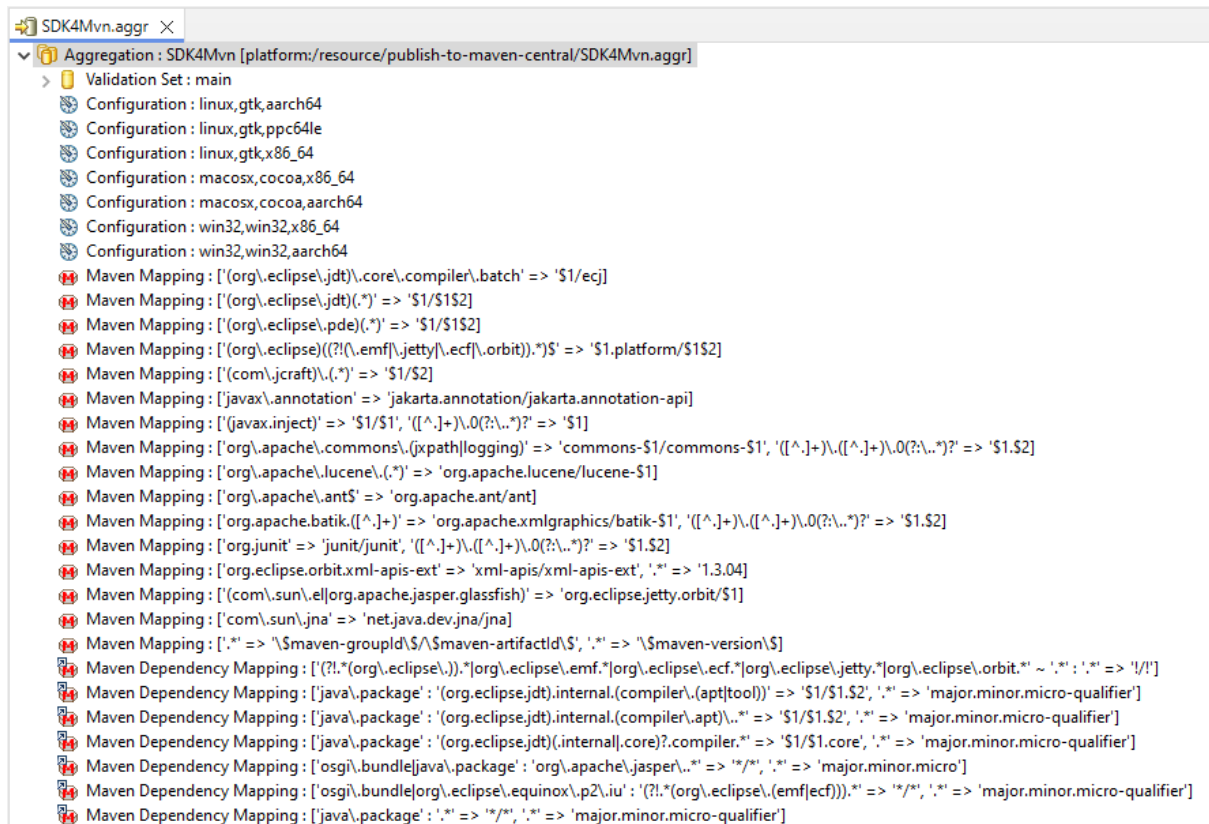
I propose to use the CBI p2 Aggregator, as already used by the Eclipse Platform and EMF, for the initial step in publishing BIRT's p2 repository to a Maven-compatible repository. This repository can then be further processed to publish the Maven artifacts, i.e., associate sources and Javadoc. An advantage of this approach is that the p2 Aggregator's tools provide analysis support for previewing the POM details in order to help detect and repair broken dependencies because it includes an analysis of what is available at Maven Central:



The screenshot shows the SDK4Mvn.aggran tool interface. The main content area displays a dependency tree for the org.eclipse.compare bundle. The tree is expanded to show the 'Compare Support' section, which lists various dependencies with their versions and Maven coordinates. The dependencies are as follows:

- org.eclipse.platform : org.eclipse.ui : 3.206.0 : bundle org.eclipse.ui [3.206.0,4.0.0) : ^3.205.100
- org.eclipse.platform : org.eclipse.core.resources : 3.20.200 : bundle org.eclipse.core.resources [3.4.0,4.0.0) : ^3.20.100
- org.eclipse.platform : org.eclipse.jface.text : 3.25.100 : bundle org.eclipse.jface.text [3.8.0,4.0.0) : ^3.25.0
- org.eclipse.platform : org.eclipse.ui.ide : 3.22.200 : bundle org.eclipse.ui.ide [3.3.0,4.0.0) : ^3.22.100
- org.eclipse.platform : org.eclipse.ui.views : 3.12.300 : bundle org.eclipse.ui.views [3.2.0,4.0.0) : ^3.12.200
- org.eclipse.platform : org.eclipse.ui.workbench.texteditor : 3.17.400 : bundle org.eclipse.ui.workbench.texteditor [3.5.0,4.0.0) : ^3.17.300
- org.eclipse.platform : org.eclipse.core.runtime : 3.31.100 : bundle org.eclipse.core.runtime [3.29.0,4.0.0) : ^3.31.0
- org.eclipse.platform : org.eclipse.core.expressions : 3.9.400 : bundle org.eclipse.core.expressions [3.2.0,4.0.0) : ^3.9.300
- org.eclipse.platform : org.eclipse.ui.editors : 3.17.300 : bundle org.eclipse.ui.editors [3.5.0,4.0.0) : ^3.17.200
- org.eclipse.platform : org.eclipse.ui.forms : 3.13.300 : bundle org.eclipse.ui.forms [3.2.0,4.0.0) : ^3.13.200
- org.eclipse.platform : org.eclipse.compare.core : 3.8.500 : bundle org.eclipse.compare.core [3.5.0,4.0.0) : ^3.8.400
- org.eclipse.platform : org.eclipse.ui.navigator : 3.12.400 : ? : bundle org.eclipse.ui.navigator [3.2.0,4.0.0) : ^3.12.300

I propose to to define rules for producing properly structured Maven coordinates and for producing properly resolving dependencies, analogous to what has been done for the Platform here:



```
SDK4Mvn.agggr X
Aggregation : SDK4Mvn [platform:/resource/publish-to-maven-central/SDK4Mvn.agggr]
  Validation Set : main
    Configuration : linux,gtk,aarch64
    Configuration : linux,gtk,ppc64le
    Configuration : linux,gtk,x86_64
    Configuration : macosx,cocoa,x86_64
    Configuration : macosx,cocoa,aarch64
    Configuration : win32,win32,x86_64
    Configuration : win32,win32,aarch64
  Maven Mapping : ['(org\.eclipse\jdt)\.core\.compiler\.batch' => '$1/ecj]
  Maven Mapping : ['(org\.eclipse\jdt)(.*)' => '$1/$1$2]
  Maven Mapping : ['(org\.eclipse\pde)(.*)' => '$1/$1$2]
  Maven Mapping : ['(org\.eclipse)((?!\.emf|\.jetty|\.ecf|\.orbit).)*$' => '$1.platform/$1$2]
  Maven Mapping : ['(com\.jcraft)(.*)' => '$1/$2]
  Maven Mapping : ['javax\.annotation' => 'jakarta.annotation/jakarta.annotation-api]
  Maven Mapping : ['javax\.inject' => '$1/$1', '([\^.]*)\.(?!\.)*' => '$1]
  Maven Mapping : ['org\.apache\.commons\.jxpath|logging' => 'commons-$1/commons-$1', '([\^.]*)\.(?!\.)*' => '$1.$2]
  Maven Mapping : ['org\.apache\.lucene(.*)' => 'org.apache.lucene/lucene-$1]
  Maven Mapping : ['org\.apache\.ant$' => 'org.apache.ant/ant]
  Maven Mapping : ['org\.apache\.batik([\^.]*)' => 'org.apache.xmlgraphics/batik-$1', '([\^.]*)\.(?!\.)*' => '$1.$2]
  Maven Mapping : ['org\.junit' => 'junit/junit', '([\^.]*)\.(?!\.)*' => '$1.$2]
  Maven Mapping : ['org\.eclipse\.orbit\.xml-apis-ext' => 'xml-apis/xml-apis-ext', '*' => '1.3.04]
  Maven Mapping : ['(com\.sun\.el|org\.apache\.jasper\.glassfish)' => 'org.eclipse.jetty.orbit/$1]
  Maven Mapping : ['com\.sun\.jna' => 'net.java.dev.jna/jna]
  Maven Mapping : ['*' => '$maven-groupId/$maven-artifactId$', '*' => '$maven-version$]
  Maven Dependency Mapping : ['(?!.*(org\.eclipse\.)*)org\.eclipse\.emf.*org\.eclipse\.ecf.*org\.eclipse\.jetty.*org\.eclipse\.orbit.* ~ '*' : '*' => '!']
  Maven Dependency Mapping : ['java\.package' : '(org\.eclipsejdt\.internal\.compiler\.(apt|tool))' => '$1/$1.$2', '*' => 'major.minor.micro-qualifier']
  Maven Dependency Mapping : ['java\.package' : '(org\.eclipsejdt\.internal\.compiler\.apt)\.?' => '$1/$1.$2', '*' => 'major.minor.micro-qualifier']
  Maven Dependency Mapping : ['java\.package' : '(org\.eclipsejdt\.(internal|core)?\.compiler.*' => '$1/$1.core', '*' => 'major.minor.micro-qualifier']
  Maven Dependency Mapping : ['osgi\.bundle|java\.package' : 'org\.apache\.jasper\.' => '*/*', '*' => 'major.minor.micro']
  Maven Dependency Mapping : ['osgi\.bundle|org\.eclipse\.equinox\.p2\.iu' : '(?!.*(org\.eclipse\.(emf|ecf)))' => '*/*', '*' => 'major.minor.micro-qualifier']
  Maven Dependency Mapping : ['java\.package' : '*' => '*/*', '*' => 'major.minor.micro-qualifier']
```

Additionally, I propose to publish SNAPSHOT builds to repo.eclipse.org.

I believe it's essential that someone actually test the published artifacts before they are pushed to Maven Central. Far too many times, problems are not reported until after the fact, and these problems, as mentioned earlier, are impossible to repair. I am hopeful that the community will be eager to help with testing:

<https://github.com/eclipse-birt/birt/issues/625>

In addition, someone needs to consider whether the artifacts I propose to publish are sufficient and are those that are actually needed. Again, I am hopeful to get constructive feedback from the community.