Sessions Configurator Guide

About Terracotta Documentation

This documentation is about Terracotta DSO, an advanced distributed-computing technology aimed at meeting special clustering requirements.

Terracotta products without the overhead and complexity of DSO meet the needs of almost all use cases and clustering requirements. To learn how to migrate from Terracotta DSO to standard Terracotta products, see Migrating From Terracotta DSO. To find documentation on non-DSO (standard) Terracotta products, see Terracotta Documentation. Terracotta release information, such as release notes and platform compatibility, is found in Product Information.

Release: 3.6
Publish Date: November, 2011 **Documentation Archive »**

Sessions Configurator Guide

Introduction

The Terracotta Sessions Configurator is a graphical tool that assists you in creating a configuration to cluster web-application session data. A `sandbox` environment is provided that consists of a Terracotta server instance and two independent web-container instances.

You can obtain the Sessions Configurator from the Terracotta forge. See the README.txt for information on how to run the Sessions Configurator.

> ⚠ The Sessions Configurator includes pre-configured web containers to run your web application. You do not need to supply or configure a separate web container.

Within the sandbox, you can exercise your webapp with Terracotta clustering enabled, taking actions when prompted to update your configuration toward the goal of clustering your session data across both web-container instances.

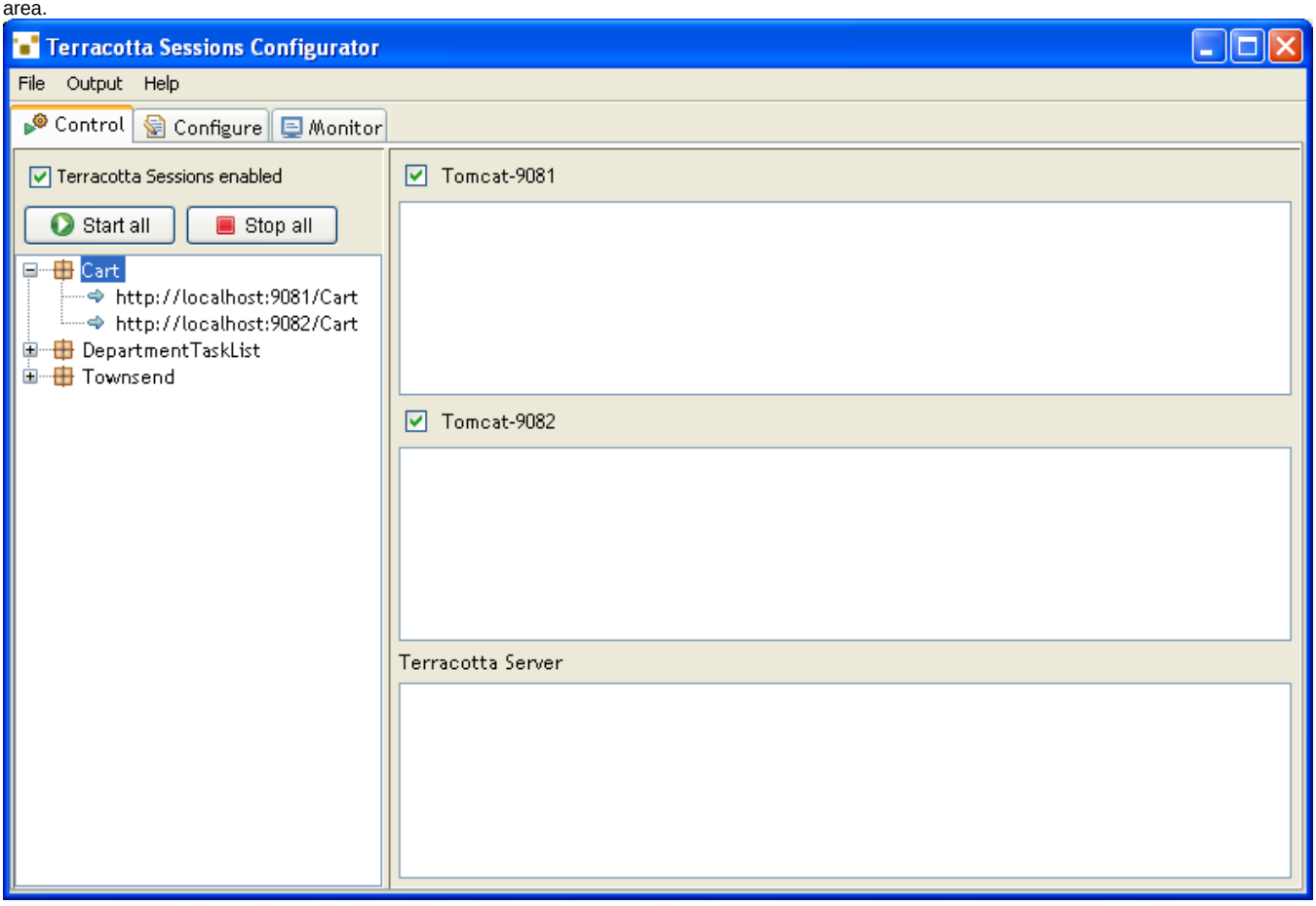
Further Reading:
See Clustering a Spring Web Application for a tutorial on using the Sessions Configurator.

Layout of The Configurator

The Sessions Configurator consists of three main panels: Control, Configure, and Monitor.

Control

The Control panel displays a list of the known webapps and links to each of the instances deployed in the two sandbox web containers. Controls allow for starting and stopping the web containers and Terracotta Server, if session clustering is enabled. The output of each server is displayed in a scrolling text area.

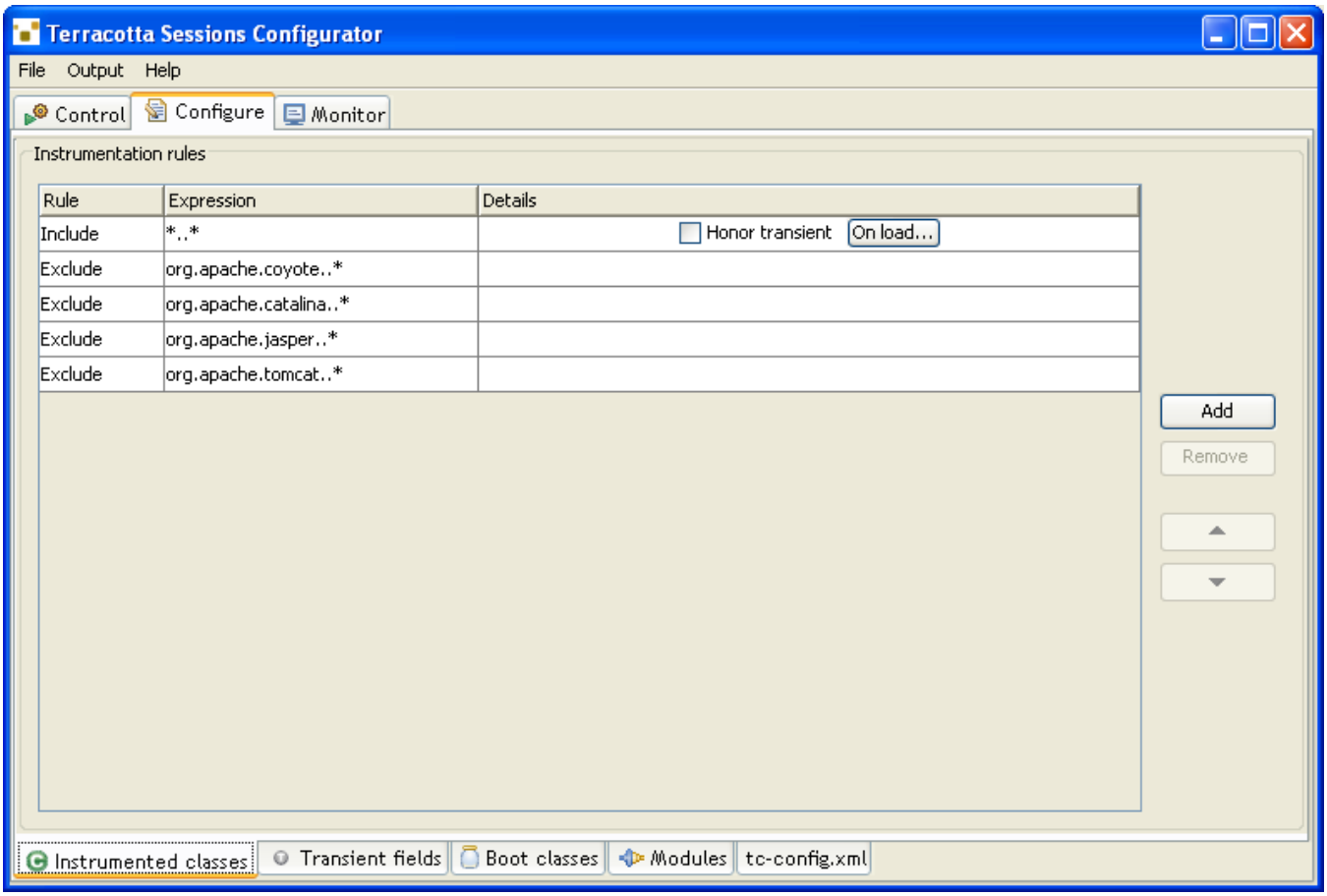


Configure

The Configure panel displays several pages relating to aspects of the Terracotta configuration file, which is displayed in the last page in raw XML form.

Instrumented classes

To cluster your webapp's session data you need to inform the Terracotta runtime of the types of data that may be added to the session. The *Instrumented classes* page lets you list patterns that describe sets of types that are to be included or excluded from the runtime instrumentation required to make a type portable.

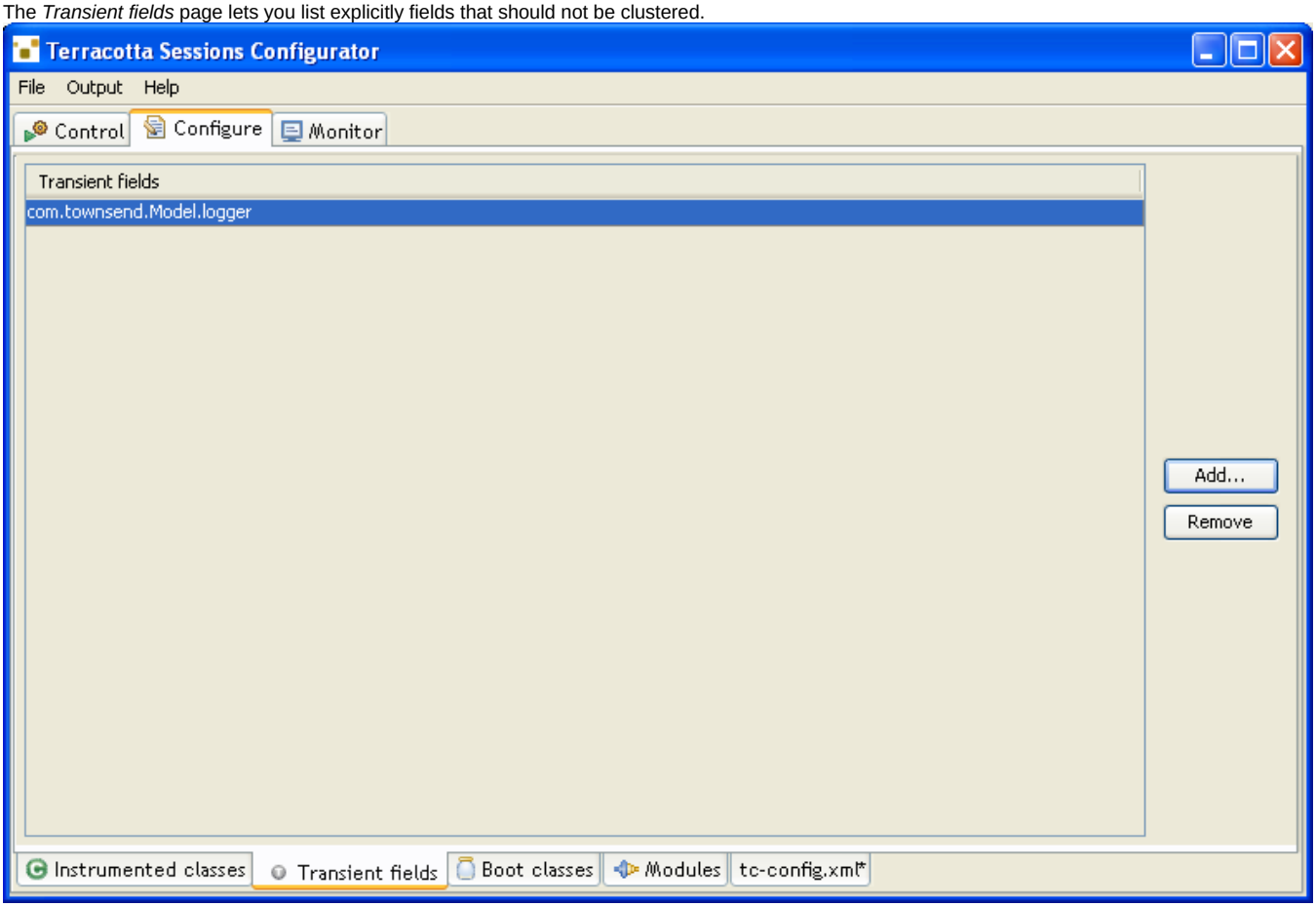


Here you can also specify more advanced treatment for a type, including whether or not fields declared as Java transient should be clustered and how fields not clustered should be initialized when their containing object is realized.

Further Reading:
For more information about object portability, see the Concept and Architecture Guide.

Transient fields

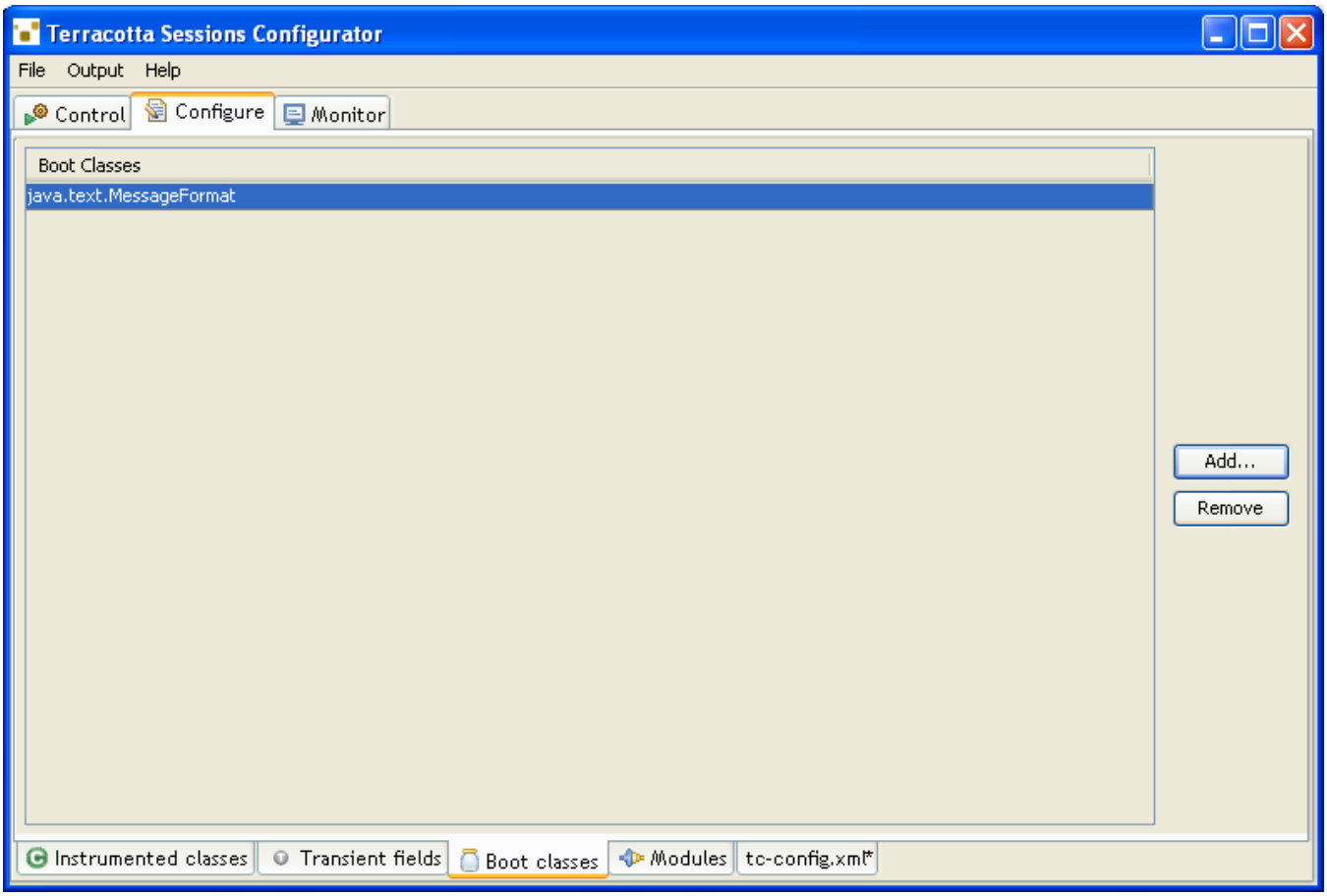
The *Transient fields* page lets you list explicitly fields that should not be clustered.



Further Reading:
For more information on handling the transient portions of a shared object, see the Concept and Architecture Guide.

Boot classes

The *Boot classes* page lets you list system types you would like clustered that are not already pre-instrumented out-of-the-box.

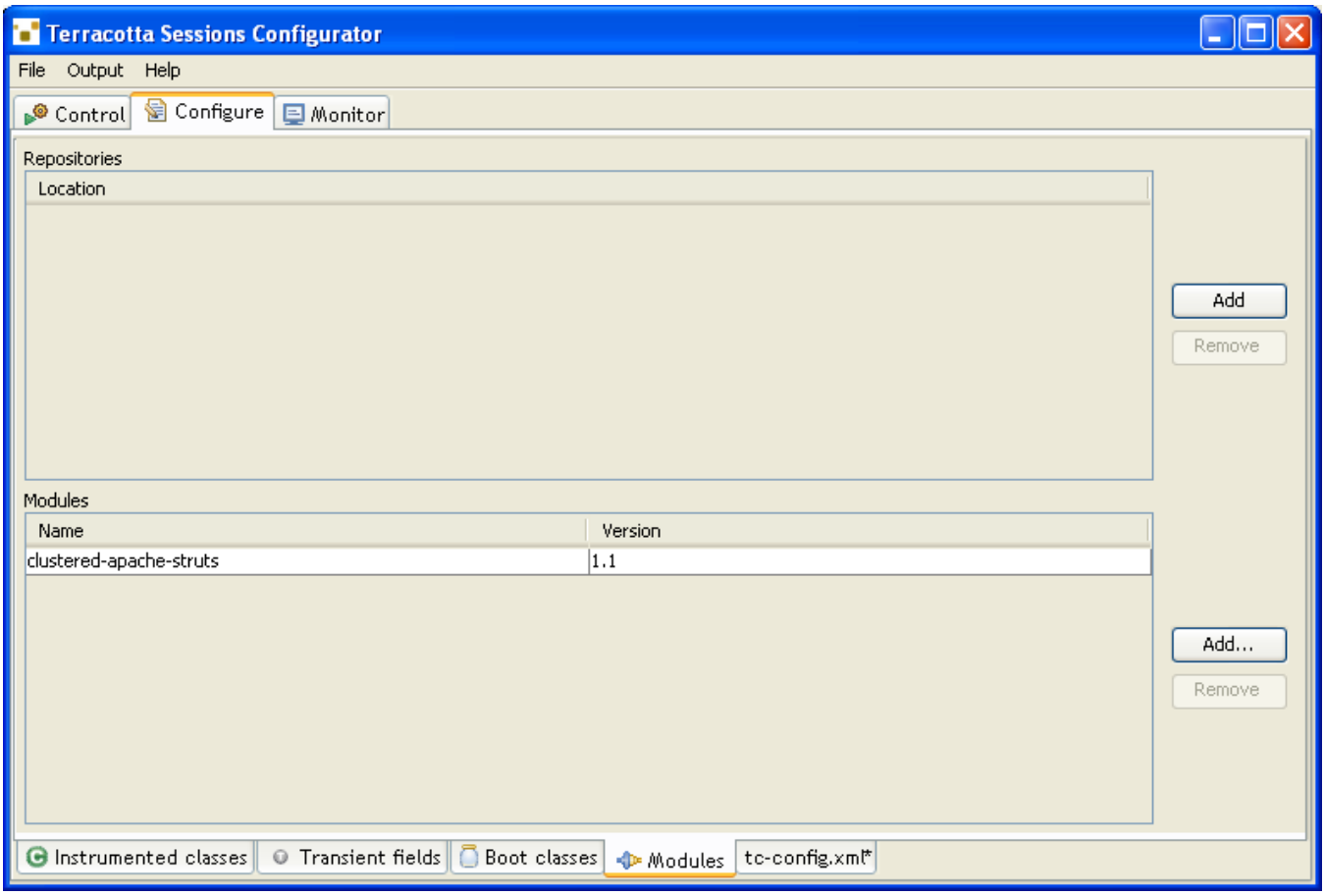


System classes must be treated specially in that Terracotta-instrumented versions of those types must be loaded via the JVM boot classpath. Types listed as boot classes will be automatically instrumented as per the details of the other pages, and included in a bootjar archive that will be loaded via the boot classpath when the web containers are started.

Further Reading:
For more information on sharing system types, see the Concept and Architecture Guide.

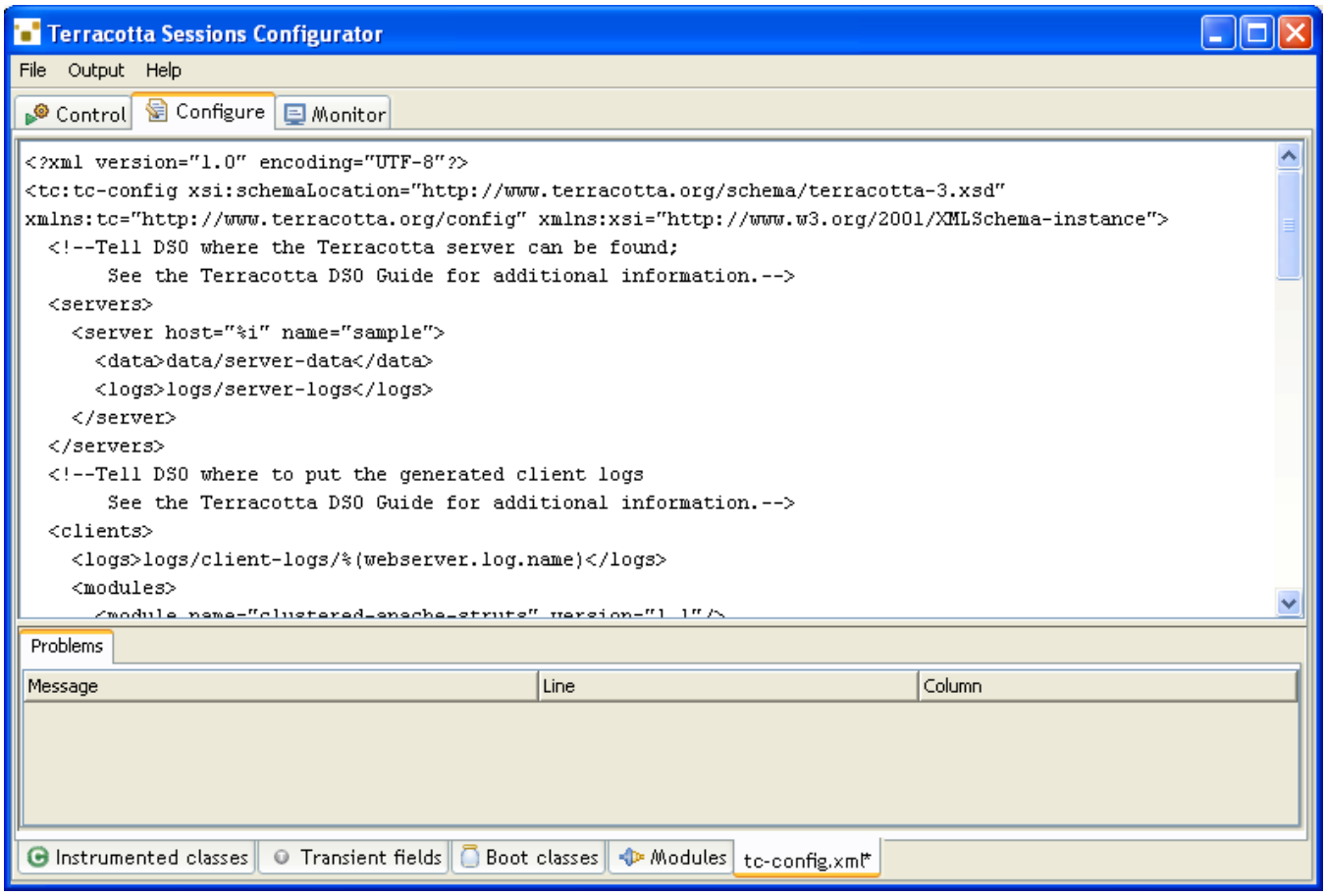
Modules

The *Modules* page lets you specify configuration modules that provide pre-packaged configuration of existing frameworks that can be used by your configuration.



tc-config.xml

The final page, `tc-config.xml` displays the raw text of the XML configuration file that is be created.

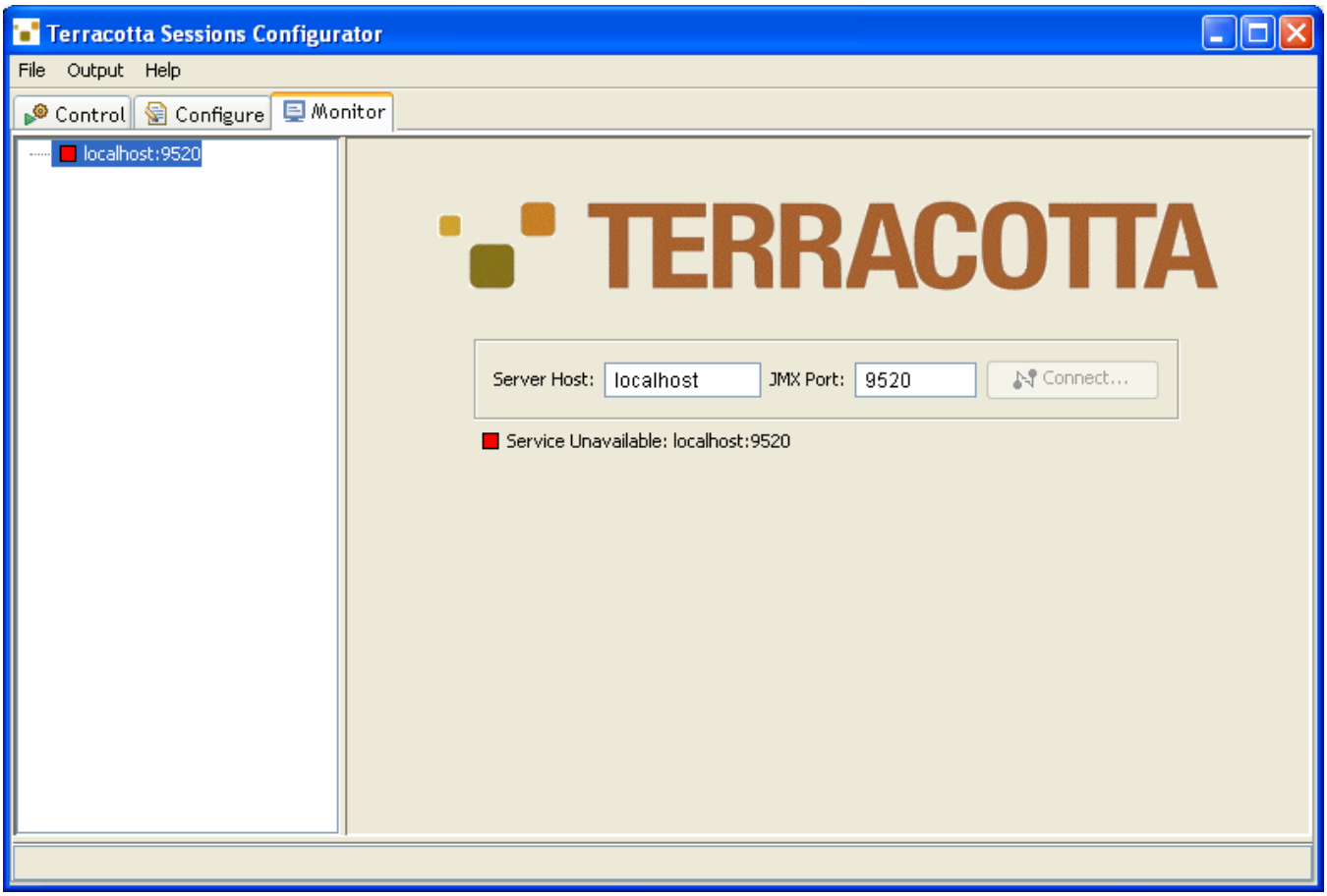


You can manually edit this file, and it will synchronize with the other pages.

Further Reading:
For more details about configuring Terracotta, see the Configuration Guide and Reference.

Monitor

The Monitor panel display an embedded instance of the Terracotta Developer Console that lets you connect to your running Terracotta Server and view various aspects of your system. Most importantly the Developer Console lets you browse your clustered session data.



Further Reading:
For more information about the Developer Console, see the Terracotta Developer Console documentation.

Importing Web Applications

Use the *File|Import webapp...* menu to browse to and import your webapp, which can either be in archived or exploded form. Standard archived webapps have a .war extension. Your webapp will be copied into the deployment area of both instances of the web container such that it will be deployed the next time the servers are started.

Verifying Your Webapp without Terracotta

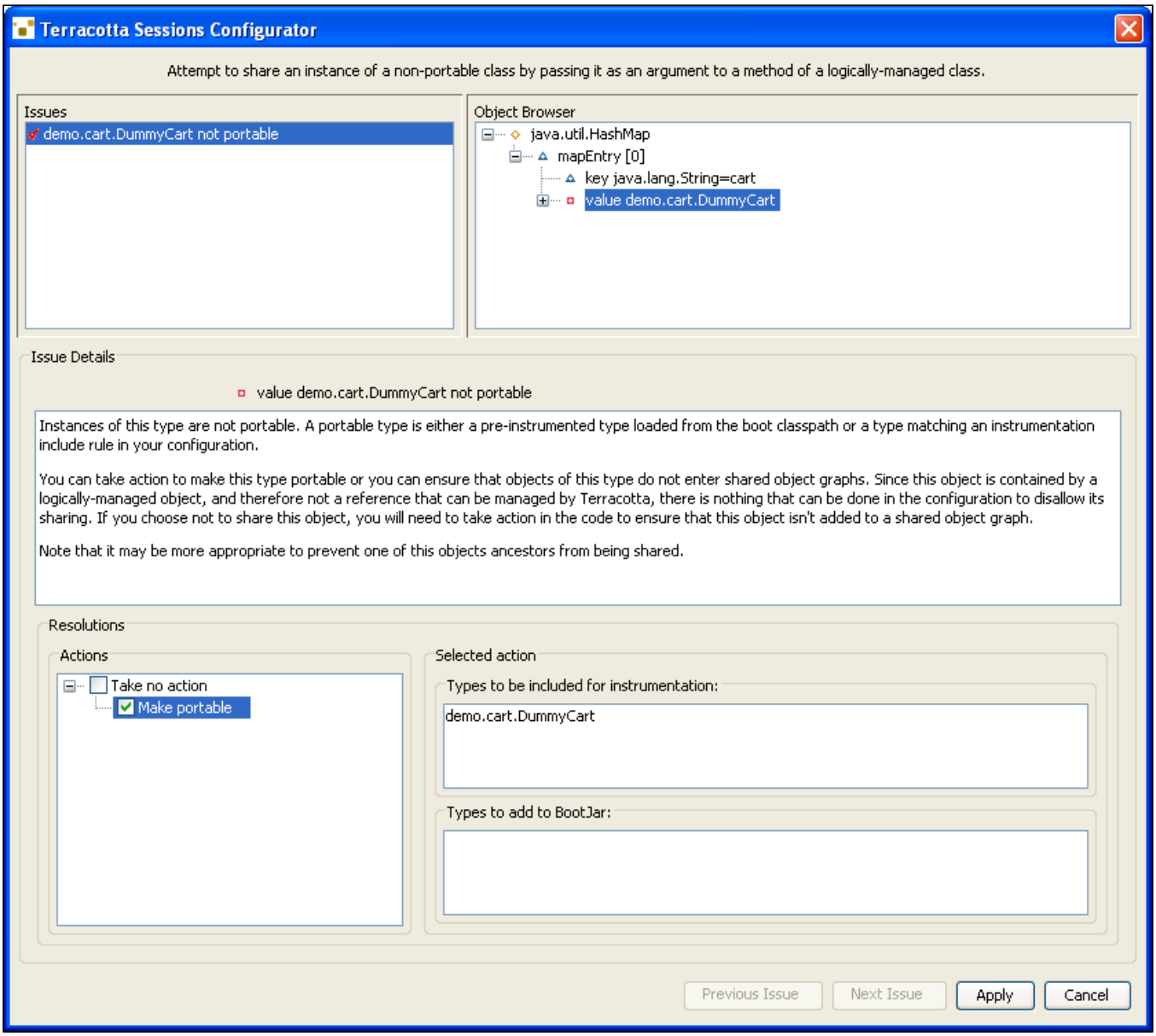
After you've successfully imported your webapp, you should verify that it works as expected without Terracotta clustering enabled. Make sure the *Terracott a enabled* checkbox on the *Control* panel is unchecked and click the *Start all* button to run both instances of your sandbox web containers. After the servers have started, the links to your webapps will become enabled and you should click them to begin validating that your webapp works as expected without Terracotta involved.

Clustering Your Webapp with Terracotta

After you're satisfied that your webapp is working properly in unclustered fashion, click the *Stop all* button to terminate the web containers. Enable Terracotta clustering and click *Start all*. First the Terracotta Server will spin up, followed by each of the web containers. Follow the links to, and begin exercising your webapp. It's quite possible at this point that you can see your clustered session data in the *Monitor* page.

Sessions Problem Feedback

When problems relating to your application configuration are encountered, such as attempts to share instances of non-portable types, a resolution dialog is shown providing an opportunity to remedy the problem by taking actions that update your Sessions configuration. The dialog displays an overview of the type of problem encountered, a list of specific issues that need to be resolved, and the application data that caused the problem. For example, if you were to remove the global instrumentations rule (`*..*`) and then try to add an item to the cart, the non-portable resolution dialog would be displayed, showing the DummyCart as being a non-portable class. You could then choose to add `DummyCart` as an instrumented-class, and your configuration file would be updated accordingly.



The types of problems encountered will be:

- attempt to share an inherently non-shareable type
- attempt to share a system type not loaded from the DSO boot jar
- attempt to share a type that has not been declared to be Terracotta instrumented

Declaring Transient Data

Any problematic object field can be excluded from sharing by declaring that field as a Terracotta **transient field.** Transient fields generally must be re-constituted when their containing object is loaded from the Terracotta Server into a DSO client application. This can be accomplished using a scripting mechanism or by arranging for a particular method to be invoked on the containing object.

A good use for Terracotta transient fields are members whose types are never appropriate to share, such as objects associated with VM-specific resources such as threads or Files. These types of objects are never shareable, with Terracotta DSO or any other clustering strategy.

For more information about specifying and handling transient data, see the Concept and Architecture Guide.

Sharing System Types

Certain system types must be loaded from the boot classpath in order to be shareable with Terracotta DSO. Many of the most commonly shared system types are automatically pre-instrumented and included in the default DSO bootjar. If other system types are required to be shared, they can be listed in the additional-boot-jar-classes section of your configuration file, and a custom bootjar will be created and used as needed. If an attempt is made to share an instance of such a system type not loaded from the DSO bootjar, the offending objects will be displayed in the non-portable resolution dialog, at which time you can choose to have them added to your custom DSO bootjar.

For more information about the DSO bootjar, see the Concept and Architecture Guide.

Adding Additional Instrumented Types

In order to share your application types, they must declared in the instrumented-classes section of your DSO configuration file. The instrumentation process occurs at load-time and allows Terracotta to inject the code required to manage your object state between the Terracotta Server and the DSO client application participating in the cluster. If an attempt is made to share an instance that was not declared for instrumentation, the offending objects will be displayed in the NonPortable resolution dialog, at which time you can choose to add those types to the set of instrumented classes in your configuration.

For more information about object portability, see the Concept and Architecture Guide.

Viewing Session Data in the Developer Console

Move to the *Monitor* page and click *Connect* to attach the Developer Console to the Terracotta Server. Browse to the Roots node and drill down to your session data. Clicking on a node causes it's children to refresh.

Moving From the Sandbox to Production

Now that you've clustered your webapp sessions, the configuration you've created can be used to move to production setting. Use the *File*|*Export configuration* menu to save the configuration to a file outsite of the sandbox.

Further Reading:
For more information about moving your Terracotta-clustered webapp to a production environment, see the Deployment Guide.