WCT Configuration and Deployment Guide

This document provides additional information on the configuration and deployment of the WCT application using one or more harvester agents. It should be read in conjunction with the following documents:

Web Curator Tool System Administrator Guide (v1.6.0 onwards).pdf
Web Curator Tool Upgrade Guide (WCT 1.6).pdf

Tomcat 5 Configuration

Disable Tag Pooling

The following changes are needed to disable tag pooling on a Tomcat 5 or 6 server. Tag pooling is enabled by default, but causes all target instances returned by a search result to be pinned within the pool, preventing them from being removed by a garbage collection.

$CATALINA_HOME\conf\web.xml

Add the line highlighted in red:

```
    <servlet>
        <servlet-name>jsp</servlet-name>
        <servlet-class>org.apache.jasper.servlet.JspServlet</servlet-
class>
        <init-param>
            <param-name>enablePooling</param-name>
            <param-value>false</param-value>
        </init-param>
        <init-param>
```

Disable PageContext Pooling/Enable Heap Shrinking

$CATALINA_HOME\bin\catalina.bat or .sh

Add the Java command line options highlighted in red:

```
…
if not exist "%CATALINA_HOME%\bin\tomcat-juli.jar" goto noJuli
set JAVA_OPTS=%JAVA_OPTS% -
Djava.util.logging.manager=org.apache.juli.ClassLoaderLogManager -
Djava.util.logging.config.file="%CATALINA_BASE%\conf\logging.properti
es"
set JAVA_OPTS=%JAVA_OPTS% -Xms64m -Xmx200m -server -
XX:MaxPermSize=200m -XX:MaxHeapFreeRatio=10 -XX:MinHeapFreeRatio=10
-XX:+CMSClassUnloadingEnabled -
Dorg.apache.jasper.runtime.JspFactoryImpl.USE_POOL=false -
Dorg.apache.jasper.runtime.BodyContentImpl.LIMIT_BUFFER=true
…
```

The CMSClassUnloadingEnabled, MaxHeapFreeRatio and MinHeapFreeRatio options above ensure that the heap will be shrunk and memory returned to the OS following a garbage collection.

Enable Clearing of Timer Threads and Threadlocal Keys

The Axis and Heritrix frameworks add ThreadLocal keys which are not cleared by Spring's lifecycle management when the application is restarted or redeployed. Furthermore, Spring does not stop scheduler threads started by the Quartz framework. These omissions cause memory leaks when the application is redeployed, and prevent WCT from shutting down cleanly.
To prevent these leaks, the following configuration is needed:

$CATALINA_HOME\conf\context.xml

Add the options highlighted in red:

```
…
<Context
      clearReferencesStopTimerThreads="true"
      clearReferencesThreadLocals="true"
>
…
```

Deployment Steps

Database Upgrade

After taking a complete backup of the database and Tomcat servers, stop WCT and upgrade the database using the document: Web Curator Tool Upgrade Guide.pdf

by following the sections entitled:

"Shut Down the WCT"

and

"Upgrading WCT Database Schema"

Application Upgrade

There are three major components to the deployment of the Web Curator Tool:

- the web curator core (wct.war)
- the web curator harvest agent (wct-harvest-agent.war)
- the web curator digital asset store (wct-store.war).

The harvest agent is normally deployed in a separate JVM from the core and digital asset store which allows it to be run on remote machines. Scalability in WCT is achieved through the addition of further harvest agents.

Deploying the war files consists of copying them into the webapps folder of the relevant Tomcat instances. Tomcat will then deploy and start the application.

The three WAR files contain a number of configuration files that need to be customised for each environment. It is recommended that copies of these files are made from the current installation before attempting a deployment. If an upgrade is being performed, these files will require a merge with the existing files to add any new configuration parameters introduced.

If configuration changes need to be applied after the wars are built, the following steps can be followed to apply them manually.

A staging directory, denoted as *STAGE* will be used to modify the war files (eg: /home/tomcat/stage/)

The deployment directory on the Tomcat instance running wct.war and wct-store.war is:

/lvdata/tomcat/webapps/

The deployment directory on the first Tomcat instance running wct-harvest-agent.war (ie: the first harvest agent) is:

/lvdata/tomcat/instances/crawler1/webapps/

1. copy the three war files to the *STAGE* directory on the target machines
2. log in to the target machine that contains wct.war file
3. cd *STAGE*
4. mkdir wct
5. cp wct.war ./wct/
6. mkdir conf
7. cd conf
8. mkdir wct
9. cd ..
10. cd wct

    if WCT is already installed on the machine then copy the existing configuration to the conf directory:

11. cp /lvdata/tomcat/webapps/wct/WEB-INF/classes/ *STAGE*/conf/wct/

    explode the war file using

12. unzip wct.war

13. cd ./WEB-INF

    The web-service configuration file contains the definition of the attachments directory used by WCT. It is crucial that this parameter is set identically in all three war files:

    <parameter name="attachments.Directory" value="/tmp/core/attachments/"/>

14. cd ./classes

    ensure the **wct-core.properties** is correct, particularly the definition of the harvest agent port:

    harvestAgent.host=mozilla.bl.uk (the host that will run the harvest listener – ie: the core)
    harvestAgent.port=8080 (the port on which this Tomcat instance is running)

    ensure that the database connection strings, username and password are correct in:

    hibernate.cfg.xml

15. cd ../..

    repack the war file using:

16. rm wct.war

17. zip –r wct.war *

    deploy the war to Tomcat by copying it to the webapps directory:

18. cp wct.war /lvdata/tomcat/webapps/

    repeat the same steps to modify the configuration of wct-store.war:

19. cd ..

20. mkdir *STAGE*/conf/wct-store/

    if WCT is already installed on the machine then copy the existing configuration to the conf directory:

21. cp /lvdata/tomcat/webapps/wct-store/WEB-INF/classes/ *STAGE*/conf/wct-store/

    explode the war using

22. cp wct-store.war ./wct-store

23. cd wct-store

24. unzip wct-store.war

25. cd ./WEB-INF

    The web-service configuration file contains the definition of the attachments directory used by WCT. It is crucial that this parameter is set identically in all three war files:

<parameter name="attachments.Directory" value="/tmp/core/attachments/"/>

26. cd ./classes

    ensure that the **wct-das.properties** is correct

27. cd ../..

    repack the war file using:

28. rm wct-store.war

29. zip –r wct-store.war *

    deploy the war to Tomcat by copying it to the webapps directory:

30. cp wct-store.war /lvdata/tomcat/webapps/

    repeat the same steps to modify the configuration of wct-harvest-agent.war:

31. cd ..

32. mkdir *STAGE*/conf/wct-harvest-agent/

    if WCT is already installed on the machine then copy the existing configuration **from the second tomcat instance** to the conf directory:

33. cp /lvdata/tomcat/instances/crawler1/webapps/wct-harvest-agent/WEB-INF/classes/ *STAGE*/conf/wct-harvest-agent/

    explode the war using

34. cp wct-harvest-agent.war ./wct-harvest-agent

35. cd wct-harvest-agent

36. unzip wct-harvest-agent.war

37. cd ./WEB-INF

    The web-service configuration file contains the definition of the attachments directory used by WCT. It is crucial that this parameter is set identically in all three war files:

    <parameter name="attachments.Directory" value="/tmp/core/attachments/"/>

38. cd ./classes

    ensure that the **wct-agent.properties** is correct, particulary:

# agent host name or ip address that the core knows about
harvestAgent.host=mozilla (the host on which wct.war is running)
# the port the agent is listening on for http connections
harvestAgent.port=18090 (the port for this Tomcat server – ie: the Tomcat instance running wct-harvester-agent.war)

39. cd ../..

    repack the war file using:

40. rm wct-harvest-agent.war

41. zip –r wct-harvest-agent.war *

    deploy the war to **the second** Tomcat instance by copying it to the webapps directory:

42. cp wct-harvest-agent.war /lvdata/tomcat/instances/crawler1/webapps/

To define further harvest agents, repeat steps 32-42 on the machine that will be running the additional harvest agents. You must ensure that you define the host and port in step 38 correctly (ie: the port for the Tomcat instance running the additional wct-harvest-agent.war).