Installing DSpace on Red Hat Enterprise Linux 5

DSpace 1.6.x

This document describes installing DSpace-1.6.2 with Redhat distribution packages which reduce much of the complexity of installation and maintenance for the System Administrator. The products Apache Maven and DSpace are exceptions and are downloaded and installed seperately.

It assumes a basic understanding of

1. bash() shell and environment variables
2. Redhat package and service management

It assumes the RHEL 5 Supplementary package channel is available, which provides Sun's Java and which requires registration with RHN, the following packages are expected to be available

1. sun java
2. apache tomcat5
3. postgresql-server
4. and ant() (a build tool)

It assumes these terms for location of source code and final destination directories

1. DS_SRC = /home/dspace/src ; DS expands to DSpace
2. DS_DST = /usr/local/dspace
3. OI_SRC = /usr/local/src – for example has apache-maven-X.Y.Z
4. OI_DST = /usr/local – OI expands to Other Installations, for example has softlink for maven
5. TMP = /var/tmp

The convention is to download to TMP, unpack to one of XX_SRC, and finally install to one of XX_DST.

While this document references Redhat, yum(8), and rpm(8) for platform and package management, as far as GNU/Linux distributions go, Debian or Arch Linux are worth mentioning. Whereas Debian and derivatives such as Ubuntu and the *idealistic* GNS have apt-get(8) for package management, Arch Linux has pacman(8). For the competent system administrator, Arch Linux is a **strong** choice.

The install procedure involves

1. installing and configuring the pre-requisite packages available from Redhat
2. creating & setting the dspace user and environment
3. installing Maven and DSpace & setting dspace.cfg TO_BE_DONE: package this for distribution
4. and finally, ensuring the ongoing behind the scenes services run & continue to run

To experiment after an initial installation, the likely procedure is to

1. stop tomcat5
2. change dspace.cfg and update by ant()
3. start tomcat5

or

1. stop tomcat5
2. remove all of DS_DST/*
3. drop the database and database user
4. create the database user and database
5. make changes to dspace.cfg and build DSpace by mvn() and ant()
6. create the DSpace administrator
7. start tomcat5

Install the pre-requisite packages available from Redhat

To install use

```
yum install java-1.6.0-sun java-1.6.0-sun-devel java-1.6.0-sun-jdbc
yum install tomcat5 tomcat5-webapps
yum install xml-commons-apis
yum install postgresql-server
yum install ant ant-apache-regexp
```

Details of Sun Java

To find the files with jvm/jre in the pathname, use

```
rpm \-ql java-1.6.0-sun |grep 'jvm/jre'
```

To show the file executed when java is called and the version number, use

```
type java
  java is /usr/bin/java
readlink /usr/bin/java
  /etc/alternatives/java
readlink /etc/alternatives/java
  /usr/lib/jvm/jre-1.6.0-sun/bin/java

/usr/lib/jvm/jre-1.6.0-sun/bin/java -version
  java version "1.6.0_21"

  Java(TM) SE Runtime Environment (build 1.6.0_21-b06)
  Java HotSpot(TM) Server VM (build 17.0-b16, mixed mode)
```

Note: set the environment variables for **JAVA_HOME** and **JRE_HOME** as described in the environment section.

```
 # to cut to the chase.. you may try this in your .bash_profile

 x='/etc/java/java.conf'
 [ -a "$x" ] && source "$x"

 x='/etc/tomcat5/tomcat5.conf'
 [ -a "$x" ] && source "$x"

 export JAVA_HOME

 # then in the shell try
 printenv |fgrep -i java

 # to obtain
 JAVA_HOME=/usr/lib/jvm/java
```

Details of Apache Tomcat

In /etc/tomcat5/tomcat5.conf, add JAVA_OPTS for UTF support and improving memory management

```
# You can pass some parameters to java
# here if you wish to
\#JAVA_OPTS="-Xminf0.1 \-Xmaxf0.3"
JAVA_OPTS="-Dfile.encoding=UTF-8 \-Xmx1G \-Xms64M"
```

In /etc/tomcat5/server.xml, insert URIEncoding="UTF-8"

```
   <Connector port="8080" URIEncoding="UTF-8" maxHttpHeaderSize="8192"
              maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
              enableLookups="false" redirectPort="8443" acceptCount="100"
              connectionTimeout="20000" disableUploadTimeout="true" />
```

In /etc/tomcat5/server.xml, insert in the HOST element the following

```
<!-- DEFINE A CONTEXT PATH FOR DSpace JSP User Interface  -->
<Context path="/jspui" docBase="/usr/local/dspace/webapps/jspui" debug="0" reloadable="true" cachingAllowed="
false" allowLinking="true"/>

<!-- DEFINE A CONTEXT PATH FOR DSpace XML User Interface (Manakin) -->
<Context path="/xmlui" docBase="/usr/local/dspace/webapps/xmlui" debug="0" reloadable="true" cachingAllowed="
false" allowLinking="true"/>

<!-- DEFINE A CONTEXT PATH FOR DSpace OAI User Interface  -->
<Context path="/oai" docBase="/usr/local/dspace/webapps/oai" debug="0" reloadable="true" cachingAllowed="false"
allowLinking="true"/>
```

To have tomcat5 start as a service

```
service tomcat5 start
```

To have tomcat5 start as an ongoing boot-time system service

```
chkconfig tomcat5 on
```

```
## The default Tomcat homepage is located at
## $CATALINA_HOME/webapps/ROOT/index.jsp

## To find it use the following commands

# updatedb
# locate index.jsp
/var/lib/tomcat5/webapps/ROOT/index.jsp
/var/lib/tomcat5/webapps/jsp-examples/security/protected/index.jsp

## CATALINA_HOME is at /var/lib/tomcat5
## where it is defined is /etc/tomcat5/tomcat5.conf

# grep -iF catalina_home /etc/tomcat5/tomcat5.conf
CATALINA_HOME="/usr/share/tomcat5"
JAVA_OPTS="$JAVA_OPTS -Dcatalina.ext.dirs=$CATALINA_HOME/shared/lib:$CATALINA_HOME/common/lib"
```

Details of Postgresql

To locate postgresql.conf

```
updatedb
locate postgresql.conf
/usr/share/pgsql/postgresql.conf.sample
```

Copy and save the sample configuration file

```
cd /usr/share/pgsql
cp postgresql.conf.sample postgresql.conf
```

Edit the file by uncommenting the **listen_addresses** line, to be sure

```
listen_addresses = 'localhost'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost', '*' = all
```

Edit pg_hba.conf to specify method of authentication other than ident

```
locate pg_hba.conf
vi /var/lib/pgsql/data/pg_hba.conf
```

1. make this the first setting

```
host dspace dspace 127.0.0.1/32 md5
```

To check the status of the postgresql service

```
service postgresql status
```

To stop the postgresql service

```
service postgresql stop
```

To start the postgresql service

```
service postgresql start
# Go ahead and start the service.
```

To have the service start at init boot time

```
chkconfig postgresql on
```

Add dspace user and set environment variables

Add dspace user account

With root user privilege, at the shell prompt, type

```
# this locates the adduser(8) command, if it is not already on your PATH
updatedb
locate adduser

# this creates the account
adduser --create-home dspace
```

Setting environment variables

Add the following in /home/dspace/.bash_profile..

```
x='/etc/java/java.conf'
[ -a "$x" ] && source "$x"

x='/etc/tomcat5/tomcat5.conf'
[ -a "$x" ] && source "$x"

export JAVA_HOME

# This value is decided after installing apache maven
export PATH=/usr/local/maven/bin:$PATH

# UNDER CONSTRUCTION
export CLASSPATH=/usr/local/dspace/lib:$CLASSPATH
```

Add dspace user to postgresql and add database

Login as user **postgres**, connect to psql as postgres, and create psql role for dspace

```
su \--login postgres
createuser \--username postgres \--no-superuser \--no-createrole \--createdb \--pwprompt dspace
Enter password for new role: xxx
Enter it again: xxx
CREATE ROLE
```

Note

1. The username:dspace and password:xxx are used in dspace.cfg below.

Login as user **dspace**, connect to psql as dspace, and create the dspace database

```
su \--login dspace
createdb \--username dspace \--encoding UNICODE \--template template0 dspace 'database for dspace'
# or, createdb \--username dspace \--encoding UTF8 \--template template0 dspace 'database for dspace'
```

To reverse the steps and start again

```
su \--login dspace \-c 'dropdb dspace'
su \--login postgres \-c 'dropuser dspace'
```

Install packages from source

Install Maven

Download apache maven to /var/tmp, unpack and install to /usr/local

```
tar xjf /var/tmp/apache-maven-x.y.z-bin.tar.bz2 \-C /usr/local/src
ln \-s /usr/local/src/apache-maven-x.y.z /usr/local/maven
```

See listing above for dspace user's PATH environment setting.

```
# /home/dspace/.bash_profile should have maven in PATH environment variable
export PATH=$PATH:/usr/local/maven/bin
```

Create this directory

```
mkdir \-p /home/dspace/.m2/repository/org/apache/maven/plugins/maven-site-plugin/
```

Put your local proxy settings in the file /home/dspace/.m2/settings.xml, the content is structured as follows

```
<settings>
<proxies>
<proxy>
<active>true</active>
<protocol>http</protocol>
<!--
<host>put.your.proxy.address.here</host>
<port>put.your.port.number.here</port>
-->
</proxy>
</proxies>
</settings>
```

Change directory to

```
cd /home/dspace/.m2/repository/org/apache/maven/plugins/maven-site-plugin/
```

From http://repo1.maven.org/maven2/org/apache/maven/plugins/maven-site-plugin get 2.0-beta-6/maven-site-plugin-2.0-beta-6.jarand get maven-metadata.xmlbut rename maven-metadata.xml as maven-metadata-central.xml and choose the site-plugin version that works best.

```
# get maven-site-plugin-2.0-beta-6.jar
# get maven-metadata.xml
mv maven-metadata.xml maven-metadata-central.xml
```

Set user dspace as owner

```
chown \-R dspace:dspace /home/dspace/.m2
```

Install DSpace

Create the **final** target installation directory

```
create /usr/local/dspace
mkdir \-p /usr/local/dspace
chown dspace:dspace /usr/local/dspace
```

From sourceforge get dspace-1.5.1-src-release.tar.bz2 or newer. Put download at /var/tmp and unpack to /home/dspace/src

Note

1. get the bz2 or gz package for unix-like platforms to avoid problems associated with line ending conventions and permissions lost in translation for other platforms

```
cd /var/tmp
ls
# you should have something like dspace-1.5.1-src-release.tar.bz2
mkdir \-p /home/dspace/src
tar xjf dspace-1.5.1-src-release.tar.bz2 \-C /home/dspace/src
chown \--recursive dspace:dspace /home/dspace/src
```

Change to user dspace at /home/dspace/src/ and create LATEST link

```
su \--login dspace
cd /home/dspace/src
ln \-s dspace-MAJOR.MINOR.PATCH-src-release LATEST
# set your own dspace version number here
```

Configure dspace.cfg

Change to config directory and make local changes to dspace.cfg

```
cd /home/dspace/src/LATEST/dspace/config
cp dspace.cfg dspace.cfg.original
vi dspace.cfg
```

Note

1. see 5.1.1. of 1.6.2 DSpace Manual for description of elements in dspace.cfg
2. compare the default values in ./dspace.cfg with mappings defined in ../pom.xml
3. dspace.dir = /usr/local/dspace
4. dspace.url = *set*
5. dspace.hostname = *set as `hostname --fqdn`*
6. dspace.name = Training DSpace at Library of Nineveh
7. db.name = postgres
8. db.username = *set* to postgresql createdb credentials
9. db.password = *as above*
10. mail.server = localhost
11. mail.from.address = *set*
12. feedback.recipient = *set*
13. mail.admin = *set*
14. alert.recipient = *set*
15. registration.notify = *set*
16. default.locale = *set*
17. default.language = *set*

```
cd /home/dspace/src/LATEST/dspace
mvn clean package > /var/tmp/mvn_clean_package_log 2>&1

# to view progress, at another terminal window, use
tail /var/tmp/mvn_clean_package_log

# at this stage, if there are warnings about missing environment variables
# you should refer to the section on setting environment variables above

# on completion, you should see

[INFO] ------------------------------------------------------------------------
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 7 minutes 48 seconds
[INFO] Finished at: Mon Jan 12 12:35:07 EST 2009
[INFO] Final Memory: 23M/51M
[INFO] ------------------------------------------------------------------------

# if the target directory does not exist
# go through the mvn_clean_package_log and look for
# any lines with the word 'missing'
# a network resource could have been unreachable

cd /home/dspace/src/LATEST/dspace/target/dspace-1.5.1-build.dir

# to initialise and install the database and DSpace
ant fresh_install

# to create the administrator account
cd /usr/local/dspace/bin
ls -l create-administrator
 -rwxrw-r-- 1 dspace dspace 2121 Jan 13 15:45 create-administrator
./create-administrator

# you then see ...
 Creating an initial administrator account
 E-mail address: [change]
 First name: [set]
 Last name: [set]
 WARNING: Password will appear on-screen.
 Password:
 Again to confirm:
 Is the above data correct? (y or n): y
 Administrator account created
```

Later, after making new changes to dspace.cfg, apply its settings across applications by the **ant init_configs** or **install_configs** methods, described below, and you may need to restart services such as tomcat5 for the settings to really apply

Note

1. See 5.1.1. of 1.6.2. Dspace Manual

```
## in the build target directory, to expose the ant switches

$ ant -p
Buildfile: build.xml

====================== DSpace platform build file ======================

Common usage:

  Fresh install, including database setup and registry loading:
    % ant fresh_install

  Update existing installation, leaving data and configuration intact:
    % ant -Dconfig=/installdir/config/dspace.cfg update

========================================================================

Main targets:

 build_webapps_wars  Compress Web Applications into .war files
 clean_database      Removes DSpace database tables, destroying data
 fresh_install       Do a fresh install of the system, overwriting any data
 load_registries     Load initial contents of registries
 setup_database      Create database tables
 update              Update installed code and web applications (without clobber
ing data/config)
 update_code         Update installed code (without clobbering data/config)
 update_webapps      Update Web Applications (without clobbering data/config)
Default target: help
```

```
 when you change DS_SRC/LATEST/dspace/config/dspace.cfg
 go to DS_SRC/LATEST/dspace/target/dspace-X.Y.Z-build.dir and
 run '''ant init_configs''' for settings to apply

 when you change DS_DST/config/dspace.cfg
 run DS_DST/bin/install_configs for settings to apply

 for settings to apply across applications you may need
 to restart processes such as tomcat5
```

Replace the javamail.jar file

After you have installed DSpace, one RHEL-provided jar needs to be manually swapped to the one provided by DSpace. If you do not, you will see "javax.mail.NoSuchProviderException" error messages whenever you attempt to send mail from DSpace:

```
cd /var/lib/tomcat5/common/lib/ && sudo rm [javamail].jar
sudo alternatives --install /usr/share/java/javamail.jar javamail /dspace/webapps/xmlui/WEB-INF/lib/mail-1.4.jar 4289
sudo alternatives --config javamail
```

Connect Tomcat5 and DSpace then TEST

To restart tomcat5

```
/etc/init.d/tomcat5 restart
```

Point your web browser to the setting for **dspace.url** in dspace.cfg

```
[[http://localhost:8080/jspui]|http://localhost:8080/jspui][[http://localhost:8080/jspui/dspace-admin]
|http://localhost:8080/jspui/dspace-admin]
```

Redirecting web-server port 80 requests to tomcat servlet

By convention, web-servers listen on port 80 to deliver content such as static html files. So that web browsers can

```
## use the more familiar url
http://www.dspace-instance.org

## instead of....
http://www.dspace-instance.org:8080/jspui
http://www.dspace-instance.org:8080/xmlui
```

- Set Tomcat to serve up DSpace by default

  ```
  cd /usr/local/tomcat/webapps
  ln -s /dspace/webapps/jspui ROOT
  ## for the Manakin interface replace jspui with xmlui
  ```

- Configure /etc/httpd/conf.d/proxy_ajp.conf

  ```
  ProxyPass  /do_not_touch  !
  ProxyPass  /  ajp://localhost:8009/
  ProxyPassReverse  /  ajp://localhost:8009/
  ```

  Note:

1. The "!" <bang> sets Apache web-server to NOT REDIRECT everything under /do_not_touch

To be revised

Media Filter for TIFF Files

In order for the filter-media script to generate thumbnails for TIFF images, the Java Advanced Imaging toolkit needs to be installed (Imaging libraries and Image I/O library - jai_codec.jar, jai_core.jar, jai_imageio.jar)

To install the JAI components, download the Java Advanced Imaging API (jre version) and the Java Advanced Imaging - Image I/O Tools from [http://java.sun.com/products/java-media/jai/current.html|http://java.sun.com/products/java-media/jai/current.html].

JAI Imaging API

To install the JRE version for Linux:

```
cd $JRE ($JRE is the path into your java runtime environment eg: /etc/alternatives/java_sdk_1.5.0/jre)
chmod u+x $downloaddir/jai-1_1_3-lib-linux-$ARCH-jre.bin
$downloaddir/jai-1_1_3-lib-linux-$ARCH-jre.bin
```

use the space bar to scroll through the licence and answer "yes" (ONLY if you agree of course...).

JAI Image I/O Tools

To install the JRE version for Linux:

- download the zip file eg: I used a stable daily build such as: http://download.java.net/media/jai-imageio/builds/daily/2008-09-02/jai-imageio-1_2-pre-dr-b04-lib-linux-i586-02_Sep_2008.zip

- unzip the I/O tools zip archive
- copy both of the extracted files (clibwrapper_jiio.jar and jai_imageio.jar) from lib/ext to your $JRE_HOME/lib/ext directory

Cron Jobs

I set up the following scheduled jobs under the dspace crontab:

```
# send out the subscription e-mails at 01:00 every day
0 1 * * * /dspace/bin/dspace sub-daily >/dev/null 2>&1

# run the media filter at 02:00 every day
0 2 * * * /dspace/bin/dspace filter-media >/dev/null 2>&1

# clean up the DSpace storage tables, 15 minutes prior to running the checksum checker
45 2 * * * /dspace/bin/dspace cleanup >/dev/null 2>&1

# run the checksum checker at 03:00 every day
0 3 * * * /dspace/bin/dspace checker -d2h -p >/dev/null 2>&1

# mail the results to the sysadmin at 06:00 every day
0 6 * * * /dspace/bin/dspace checker-emailer -c >/dev/null 2>&1

# rebuild the index at 01:15 every day
15 1 * * * (cd /dspace && /dspace/bin/dspace index-init && /dspace/bin/dspace index-update) >/dev/null 2>&1

# run the item counter every hour
05 * * * * /dspace/bin/dspace itemcounter >/dev/null 2>&1

# rebuild the sitemap every day
0 6 * * * /dspace/bin/dspace generate-sitemaps >/dev/null 2>&1

# run the embargo lifter every day
15 6 * * * /dspace/bin/dspace embargo-lifter >/dev/null 2>&1

# statistics #

# build general stats every day at 1:30
30 1 * * * /dspace/bin/dspace stat-general >/dev/null 2>&1

# build monthly stats the first day of every month
40 1 1 * * /dspace/bin/dpace stat-monthly >/dev/null 2>&1

# run general stat reports every morning
05 7 * * * /dspace/bin/dspace stat-report-general >/dev/null 2>&1

# run monthly stat reports the first day of every month
10 7 1 * * /dspace/bin/dspace stat-report-monthly >/dev/null 2>
```

and the following under the root crontab:

```
######################################################################
#                                                                    #
# DSpace cron jobs (the rest are in dspace user's crontab       #
#                                                                    #
######################################################################

# Clean up the database nightly at 4.20am
20 4 * * * vacuumdb --analyze dspace > /dev/null 2>&1

# Carry out dspace snapshot at the lowest traffic period of the day
05 5 * * * /root/scripts/dspace_backup.sh
```

DSpace 1.5.x

This document describes installing DSpace-1.5.1. with Redhat distribution packages which reduce much of the complexity of installation and maintenance for the System Administrator. The products Apache Maven and DSpace are exceptions and are downloaded and installed seperately.

It assumes a basic understanding of

1. bash() shell and environment variables
2. Redhat package and service management

It assumes the RHEL 5 Supplementary package channel is available, which provides Sun's Java and which requires registration with RHN, the following packages are expected to be available

1. sun java
2. apache tomcat5
3. postgresql-server
4. and ant() (a build tool)

It assumes these terms for location of source code and final destination directories

1. DS_SRC = /home/dspace/src ; DS expands to DSpace
2. DS_DST = /usr/local/dspace
3. OI_SRC = /usr/local/src – for example has apache-maven-X.Y.Z
4. OI_DST = /usr/local – OI expands to Other Installations, for example has softlink for maven
5. TMP = /var/tmp

The convention is to download to TMP, unpack to one of XX_SRC, and finally install to one of XX_DST.

While this document references Redhat, yum(8), and rpm(8) for platform and package management, as far as GNU/Linux distributions go, Debian or Arch Linux are worth mentioning. Whereas Debian and derivatives such as Ubuntu and the *idealistic* GNS have apt-get(8) for package management, Arch Linux has pacman(8). For the competent system administrator, Arch Linux is a **strong** choice.

The install procedure involves

1. installing and configuring the pre-requisite packages available from Redhat
2. creating & setting the dspace user and environment
3. installing Maven and DSpace & setting dspace.cfg TO_BE_DONE: package this for distribution
4. and finally, ensuring the ongoing behind the scenes services run & continue to run

To experiment after an initial installation, the likely procedure is to

1. stop tomcat5
2. change dspace.cfg and update by ant()
3. start tomcat5

or

1. stop tomcat5
2. remove all of DS_DST/*
3. drop the database and database user
4. create the database user and database
5. make changes to dspace.cfg and build DSpace by mvn() and ant()
6. create the DSpace administrator
7. start tomcat5

Install the pre-requisite packages available from Redhat

To install use

```
yum install java-1.5.0-sun java-1.5.0-sun-devel
yum install tomcat5 tomcat5-webapps
yum install postgresql-server
yum install ant ant-apache-regexp
```

Details of Sun Java

To find the files with jvm/jre in the pathname, use

```
rpm \-ql java-1.5.0-sun |grep 'jvm/jre'
```

To show the file executed when java is called and the version number, use

```
type java
  java is /usr/bin/java
readlink /usr/bin/java
  /etc/alternatives/java
readlink /etc/alternatives/java
  /usr/lib/jvm/jre-1.5.0-sun/bin/java

/usr/lib/jvm/jre-1.5.0-sun/bin/java -version
  java version "1.5.0_17"
  Java(TM) 2 Runtime Environment, Standard Edition (build 1.5.0_17-b04)
  Java HotSpot(TM) Client VM (build 1.5.0_17-b04, mixed mode)
```

Note: set the environment variables for **JAVA_HOME** and **JRE_HOME** as described in the environment section.

```
 # to cut to the chase.. you may try this in your .bash_profile

 x='/etc/java/java.conf'
 [ -a "$x" ] && source "$x"

 x='/etc/tomcat5/tomcat5.conf'
 [ -a "$x" ] && source "$x"

 export JAVA_HOME

 # then in the shell try
 printenv |fgrep -i java

 # to obtain
 JAVA_HOME=/usr/lib/jvm/java
```

Details of Apache Tomcat

In /etc/tomcat5/tomcat5.conf, add JAVA_OPTS for UTF support and improving memory management

```
# You can pass some parameters to java
# here if you wish to
\#JAVA_OPTS="-Xminf0.1 \-Xmaxf0.3"
JAVA_OPTS="-Dfile.encoding=UTF-8 \-Xmx1G \-Xms64M"
```

In /etc/tomcat5/server.xml, insert URIEncoding="UTF-8"

```
   <Connector port="8080" URIEncoding="UTF-8" maxHttpHeaderSize="8192"
              maxThreads="150" minSpareThreads="25" maxSpareThreads="75"
              enableLookups="false" redirectPort="8443" acceptCount="100"
              connectionTimeout="20000" disableUploadTimeout="true" />
```

In /etc/tomcat5/server.xml, insert in the HOST element the following

```
<!-- DEFINE A CONTEXT PATH FOR DSpace JSP User Interface  -->
<Context path="/jspui" docBase="/usr/local/dspace/webapps/jspui" debug="0" reloadable="true" cachingAllowed="
false" allowLinking="true"/>

<!-- DEFINE A CONTEXT PATH FOR DSpace XML User Interface (Manakin) -->
<Context path="/xmlui" docBase="/usr/local/dspace/webapps/xmlui" debug="0" reloadable="true" cachingAllowed="
false" allowLinking="true"/>

<!-- DEFINE A CONTEXT PATH FOR DSpace OAI User Interface  -->
<Context path="/oai" docBase="/usr/local/dspace/webapps/oai" debug="0" reloadable="true" cachingAllowed="false"
allowLinking="true"/>
```

To have tomcat5 start as a service

```
service tomcat5 start
```

To have tomcat5 start as an ongoing boot-time system service

```
chkconfig tomcat5 on
```

```
## The default Tomcat homepage is located at
## $CATALINA_HOME/webapps/ROOT/index.jsp

## To find it use the following commands

# updatedb
# locate index.jsp
/var/lib/tomcat5/webapps/ROOT/index.jsp
/var/lib/tomcat5/webapps/jsp-examples/security/protected/index.jsp

## CATALINA_HOME is at /var/lib/tomcat5
## where it is defined is /etc/tomcat5/tomcat5.conf

# grep -iF catalina_home /etc/tomcat5/tomcat5.conf
CATALINA_HOME="/usr/share/tomcat5"
JAVA_OPTS="$JAVA_OPTS -Dcatalina.ext.dirs=$CATALINA_HOME/shared/lib:$CATALINA_HOME/common/lib"
```

Details of Postgresql

To locate postgresql.conf

```
updatedb
locate postgresql.conf
/usr/share/pgsql/postgresql.conf.sample
```

Copy and save the sample configuration file

```
cd /usr/share/pgsql
cp postgresql.conf.sample postgresql.conf
```

Edit the file by uncommenting the **listen_addresses** line, to be sure

```
listen_addresses = 'localhost'          # what IP address(es) to listen on;
                                        # comma-separated list of addresses;
                                        # defaults to 'localhost', '*' = all
```

Edit pg_hba.conf to specify method of authentication other than ident

```
locate pg_hba.conf
vi /var/lib/pgsql/data/pg_hba.conf
```

1. make this the first setting

```
host  dspace  dspace  127.0.0.1/32  md5
```

To check the status of the postgresql service

```
service postgresql status
```

To stop the postgresql service

```
service postgresql stop
```

To start the postgresql service

```
service postgresql start
# Go ahead and start the service.
```

To have the service start at init boot time

```
chkconfig postgresql on
```

Add dspace user and set environment variables

Add dspace user account

With root user privilege, at the shell prompt, type

```
# this locates the adduser(8) command, if it is not already on your PATH
updatedb
locate adduser

# this creates the account
adduser --create-home dspace
```

Setting environment variables

Add the following in /home/dspace/.bash_profile..

```
x='/etc/java/java.conf'
[ -a "$x" ] && source "$x"

x='/etc/tomcat5/tomcat5.conf'
[ -a "$x" ] && source "$x"

export JAVA_HOME

# This value is decided after installing apache maven
export PATH=/usr/local/maven/bin:$PATH

# UNDER CONSTRUCTION
export CLASSPATH=/usr/local/dspace/lib:$CLASSPATH
```

Add dspace user to postgresql and add database

Login as user **postgres**, connect to psql as postgres, and create psql role for dspace

```
su \--login postgres
createuser \--username postgres \--no-superuser \--no-createrole \--createdb \--pwprompt dspace
Enter password for new role: xxx
Enter it again: xxx
CREATE ROLE
```

Note

1. The username:dspace and password:xxx are used in dspace.cfg below.

Login as user **dspace**, connect to psql as dspace, and create the dspace database

```
su \--login dspace
createdb \--username dspace \--encoding UNICODE \--template template0 dspace 'database for dspace'
# or, createdb \--username dspace \--encoding UTF8 \--template template0 dspace 'database for dspace'
```

To reverse the steps and start again

```
su \--login dspace \-c 'dropdb dspace'
su \--login postgres \-c 'dropuser dspace'
```

Install packages from source

Install Maven

Download apache maven to /var/tmp, unpack and install to /usr/local

```
tar xjf /var/tmp/apache-maven-x.y.z-bin.tar.bz2 \-C /usr/local/src
ln \-s /usr/local/src/apache-maven-x.y.z /usr/local/maven
```

See listing above for dspace user's PATH environment setting.

```
# /home/dspace/.bash_profile should have maven in PATH environment variable
export PATH=$PATH:/usr/local/maven/bin
```

Create this directory

```
mkdir \-p /home/dspace/.m2/repository/org/apache/maven/plugins/maven-site-plugin/
```

Put your local proxy settings in the file /home/dspace/.m2/settings.xml, the content is structured as follows

```
<settings>
<proxies>
<proxy>
<active>true</active>
<protocol>http</protocol>
<!--
<host>put.your.proxy.address.here</host>
<port>put.your.port.number.here</port>
-->
</proxy>
</proxies>
</settings>
```

Change directory to

```
cd /home/dspace/.m2/repository/org/apache/maven/plugins/maven-site-plugin/
```

From http://repo1.maven.org/maven2/org/apache/maven/plugins/maven-site-plugin

get 2.0-beta-6/maven-site-plugin-2.0-beta-6.jar

and get maven-metadata.xml

but rename maven-metadata.xml as maven-metadata-central.xml and choose the site-plugin version that works best.

```
# get maven-site-plugin-2.0-beta-6.jar
# get maven-metadata.xml
mv maven-metadata.xml maven-metadata-central.xml
```

Set user dspace as owner

```
chown \-R dspace:dspace /home/dspace/.m2
```

Install DSpace

Create the **final** target installation directory

```
create /usr/local/dspace
mkdir \-p /usr/local/dspace
chown dspace:dspace /usr/local/dspace
```

From sourceforge get dspace-1.5.1-src-release.tar.bz2 or newer. Put download at /var/tmp and unpack to /home/dspace/src

Note

1. get the bz2 or gz package for unix-like platforms to avoid problems associated with line ending conventions and permissions lost in translation for other platforms

```
cd /var/tmp
ls
# you should have something like dspace-1.5.1-src-release.tar.bz2
mkdir \-p /home/dspace/src
tar xjf dspace-1.5.1-src-release.tar.bz2 \-C /home/dspace/src
chown \--recursive dspace:dspace /home/dspace/src
```

Change to user dspace at /home/dspace/src/ and create LATEST link

```
su \--login dspace
cd /home/dspace/src
ln \-s dspace-MAJOR.MINOR.PATCH-src-release LATEST
# set your own dspace version number here
```

Configure dspace.cfg

Change to config directory and make local changes to dspace.cfg

```
cd /home/dspace/src/LATEST/dspace/config
cp dspace.cfg dspace.cfg.original
vi dspace.cfg
```

Note

1. see 5.1.1. of 1.5.1 DSpace Manual for description of elements in dspace.cfg
2. compare the default values in ./dspace.cfg with mappings defined in ../pom.xml
3. dspace.dir = /usr/local/dspace
4. dspace.url = set
5. dspace.hostname = set as `hostname --fqdn`
6. dspace.name = Training DSpace at Library of Nineveh
7. db.name = postgres
8. db.username = set to postgresql createdb credentials
9. db.password = as above
10. mail.server = localhost
11. mail.from.address = set
12. feedback.recipient = set
13. mail.admin = set
14. alert.recipient = set
15. registration.notify = set
16. default.locale = set
17. default.language = set

```
 cd /home/dspace/src/LATEST/dspace
 mvn clean package > /var/tmp/mvn_clean_package_log 2>&1

 # to view progress, at another terminal window, use
 tail /var/tmp/mvn_clean_package_log

 # at this stage, if there are warnings about missing environment variables
 # you should refer to the section on setting environment variables above

 # on completion, you should see

[INFO] ------------------------------------------------------------------------
[INFO] ------------------------------------------------------------------------
[INFO] BUILD SUCCESSFUL
[INFO] ------------------------------------------------------------------------
[INFO] Total time: 7 minutes 48 seconds
[INFO] Finished at: Mon Jan 12 12:35:07 EST 2009
[INFO] Final Memory: 23M/51M
[INFO] ------------------------------------------------------------------------

 # if the target directory does not exist
 # go through the mvn_clean_package_log and look for
 # any lines with the word 'missing'
 # a network resource could have been unreachable

 cd /home/dspace/src/LATEST/dspace/target/dspace-1.5.1-build.dir

 # to initialise and install the database and DSpace
 ant fresh_install

 # to create the administrator account
 cd /usr/local/dspace/bin
 ls -l create-administrator
  -rwxrw-r-- 1 dspace dspace 2121 Jan 13 15:45 create-administrator
 ./create-administrator

 # you then see ...
  Creating an initial administrator account
  E-mail address: [change]
  First name: [set]
  Last name: [set]
  WARNING: Password will appear on-screen.
  Password:
  Again to confirm:
  Is the above data correct? (y or n): y
  Administrator account created
```

Later, after making new changes to dspace.cfg, apply its settings across applications by the **ant init_configs** or **install_configs** methods, described below, and you may need to restart services such as tomcat5 for the settings to really apply

Note

1. See 5.1.1. of 1.5.1. Dspace Manual

```
## in the build target directory, to expose the ant switches

$ ant -p
Buildfile: build.xml

====================== DSpace platform build file ======================

Common usage:

  Fresh install, including database setup and registry loading:
    % ant fresh_install

  Update existing installation, leaving data and configuration intact:
    % ant -Dconfig=/installdir/config/dspace.cfg update

=============================================================================

Main targets:

 build_webapps_wars  Compress Web Applications into .war files
 clean_database      Removes DSpace database tables, destroying data
 fresh_install       Do a fresh install of the system, overwriting any data
 load_registries     Load initial contents of registries
 setup_database      Create database tables
 update              Update installed code and web applications (without clobber
ing data/config)
 update_code         Update installed code (without clobbering data/config)
 update_webapps      Update Web Applications (without clobbering data/config)
Default target: help
```

```
 when you change DS_SRC/LATEST/dspace/config/dspace.cfg
 go to DS_SRC/LATEST/dspace/target/dspace-X.Y.Z-build.dir and
 run '''ant init_configs''' for settings to apply

 when you change DS_DST/config/dspace.cfg
 run DS_DST/bin/install_configs for settings to apply

 for settings to apply across applications you may need
 to restart processes such as tomcat5
```

Connect Tomcat5 and DSpace then TEST

To restart tomcat5

```
/etc/init.d/tomcat5 restart
```

Point your web browser to the setting for **dspace.url** in dspace.cfg

```
 [http://localhost:8080/jspui]
[http://localhost:8080/jspui/dspace-admin]
```

Redirecting web-server port 80 requests to tomcat servlet

By convention, web-servers listen on port 80 to deliver content such as static html files. So that web browsers can

```
## use the more familiar url
http://www.dspace-instance.org

## instead of....
http://www.dspace-instance.org:8080/jspui
http://www.dspace-instance.org:8080/xmlui
```

- Set Tomcat to serve up DSpace by default

```
cd /usr/local/tomcat/webapps
ln -s /dspace/webapps/jspui ROOT
## for the Manakin interface replace jspui with xmlui
```

- Configure /etc/httpd/conf.d/proxy_ajp.conf

```
ProxyPass  /do_not_touch  !
ProxyPass  /  ajp://localhost:8009/
ProxyPassReverse  /  ajp://localhost:8009/
```

Note:

1. The "!" <bang> sets Apache web-server to NOT REDIRECT everything under /do_not_touch

To be revised

Media Filter for TIFF Files

In order for the filter-media script to generate thumbnails for TIFF images, the Java Advanced Imaging toolkit needs to be installed (Imaging libraries and Image I/O library - jai_codec.jar, jai_core.jar, jai_imageio.jar)

To install the JAI components, download the Java Advanced Imaging API (jre version) and the Java Advanced Imaging - Image I/O Tools from http://java.sun.com/products/java-media/jai/current.html.

JAI Imaging API

To install the JRE version for Linux:

```
cd $JRE ($JRE is the path into your java runtime environment eg: /etc/alternatives/java_sdk_1.5.0/jre)
chmod u+x $downloaddir/jai-1_1_3-lib-linux-$ARCH-jre.bin
$downloaddir/jai-1_1_3-lib-linux-$ARCH-jre.bin
```

use the space bar to scroll through the licence and answer "yes" (ONLY if you agree of course...).

JAI Image I/O Tools

To install the JRE version for Linux:

- download the zip file eg: I used a stable daily build such as:http://download.java.net/media/jai-imageio/builds/daily/2008-09-02/jai-imageio-1_2-pre-dr-b04-lib-linux-i586-02_Sep_2008.zip
- unzip the I/O tools zip archive
- copy both of the extracted files (clibwrapper_jiio.jar and jai_imageio.jar) from lib/ext to your $JRE_HOME/lib/ext directory

Cron Jobs

I set up the following scheduled jobs under the dspace crontab:

```
# Send out subscription e-mails at midnight every day
0 0 * * * /dspace/bin/sub-daily

# Run the media filter at 0:20 every day
20 0 * * * /dspace/bin/filter-media -n

# Run stat analyses
0 1 * * * /dspace/bin/stat-general
5 1 * * * /dspace/bin/stat-monthly
0 2 * * * /dspace/bin/stat-report-general
5 2 * * * /dspace/bin/stat-report-monthly

# Run the checksum checker at 03:00
0 3 * * * /dspace/bin/checker -lp

# Run the indexer
30 3 * * * /dspace/bin/index-update

# Mail the results to the sysadmin at 04:00
0 4 * * * /dspace/bin/dsrun org.dspace.checker.DailyReportEmailer -c

# Collection strengths:
0 5 * * * /dspace/bin/dsrun org.dspace.browse.ItemCounter

# Generate sitemaps
#0 6 * * * /dspace/bin/generate-sitemaps
```

and the following under the root crontab:

```
######################################################################
#                                                                    #
# DSpace cron jobs (the rest are in dspace user's crontab       #
#                                                                    #
######################################################################

# Clean up the database nightly at 4.20am
20 4 * * * vacuumdb --analyze dspace > /dev/null 2>&1

# Carry out dspace snapshot at the lowest traffic period of the day
05 5 * * * /root/scripts/dspace_backup.sh
```

DSpace 1.4.x

The aim of this method of installing DSpace is to minimise the maintenance burden on the system administrator by using Red Hat-supplied packages where possible. This includes making use of the RHEL 5 *Supplementary* channel for the Sun Java 1.5.0 package.

Install the prerequisite Red Hat packages

- Make sure your system is subscribed to the Supplementary channel.
- Install all the required packages that can be obtained from Red Hat
    - ant
    - ant-apache-regexp
    - httpd
    - java-1.5.0-sun-devel
    - postgresql-server
    - postgresql-jdbc

```
yum \-y install ant ant-apache-regexp httpd java-1.5.0-sun-devel postgresql-server postgresql-jdbc
```

Create a DSpace User

Create a unix account for the dspace process.

```
useradd dspace
```

There's no need to give it login ability.

Tomcat 5.5

Unfortunately, the version of Tomcat 5 that is packaged with RHEL5 does not work cleanly with the proprietary Javas (including Sun's). Fortunately, the binary distribution of Tomcat 5.5 that you can download from the Tomcat website installs and runs cleanly.

Download

Grab the tarball of the **Core** version of Tomcat 5.5 [download site|http://tomcat.apache.org/download-55.cgi] and put it in /tmp directory.

Install

Change to the dspace user, copy it to the dspace user's home directory and extract it.

```
[privuser@dhost ~]$ sudo su - dspace
[dspace@dhost ~]$ cp /tmp/apache-tomcat-5.5.26.tar.gz ./
[dspace@dhost ~]$ tar zxf apache-tomcat-5.5.26.tar.gz
[dspace@dhost ~]$ exit
[privuser@dhost ~]$
```

As root, move the extracted apache-tomcat-5.5.x directory to /usr/local/

```
[privuser@dhost ~]$ sudo mv /home/dspace/apache-tomcat-5.5.26 /usr/local/
[privuser@dhost ~]$
```

Create a symlink from /usr/local/tomcat to this directory.

```
[privuser@dhost ~]$ sudo ln -s /usr/local/apache-tomcat-5.5.26 /usr/local/tomcat
[privuser@dhost ~]$
```

Set up the Enviroment Variables

Create the file **/etc/profile.d/dspace.sh** and set required the environment variables in it. Using a separate profile.d file is neater than modifying the /etc /profile file directly.

```
#!/bin/bash

export CATALINA_HOME=/usr/local/tomcat
export CATALINA_BASE=/usr/local/tomcat
export CATALINA_TMPDIR=/usr/local/tomcat/temp
if [ `/bin/arch` = "x86_64" ]; then
        XARCH=".x86_64"
fi
export JRE_HOME=/usr/lib/jvm/jre-1.5.0-sun${XARCH}
export JAVA_HOME=/usr/lib/jvm/java-1.5.0-sun${XARCH}
```

Install init.d Script

Create **/etc/rc.d/init.d/dspace** with the following contents:

```
#!/bin/bash
############################################################
# chkconfig: 345 80 20                                     #
# Description: This is an init.d script for dspace/tomcat #
############################################################

# set environment

[ -n "${TOMCAT_USER}" ] || export TOMCAT_USER=dspace
[ -n "${CATALINA_BASE}" ] || export CATALINA_BASE=/usr/local/tomcat
[ -n "${CATALINA_HOME}" ] || export CATALINA_HOME=/usr/local/tomcat
[ -n "${CATALINA_TMPDIR}" ] || export CATALINA_TMPDIR=/usr/local/tomcat/temp
# Need to specify a different java path if we're x86_64
if [ `/bin/arch` = "x86_64" ]; then
        XARCH=".x86_64"
fi
[ -n "${JAVA_HOME}" ] || export JAVA_HOME=/usr/lib/jvm/java-1.5.0-sun${XARCH}
[ -n "${JRE_HOME}" ] || export JRE_HOME=/usr/lib/jvm/jre-1.5.0-sun${XARCH}
export JAVA_OPTS="-Xmx800m -Xms800m -Dcatalina.base=$CATALINA_BASE"

GREP_STRING="dspace"

PROG=$(basename $0)

# For SELinux we need to use 'runuser' not 'su'
if [ -x "/sbin/runuser" ]; then
    SU="/sbin/runuser"
else
    SU="su"
fi

stop() {
        echo "Stopping tomcat instance $PROG..."
        $CATALINA_HOME/bin/catalina.sh stop
        echo "Done."
}

status() {
        PID=`ps -ef | grep catalina.start | grep ${GREP_STRING} | awk '!/grep/ {
print $2}'`
        if [[ -n "$PID" ]]; then
                echo $"${PROG} is running (pid $PID)."
        else
                echo $"${PROG} is not running..."
                echo ""
        fi
}


start() {
        echo "Starting tomcat instance $PROG..."
        ${SU} -m ${TOMCAT_USER} -c "$CATALINA_HOME/bin/catalina.sh start"
        echo "Done."
}

restart() {
        stop
        echo "Pausing 15 seconds before restarting."
        sleep 15
        start
}

case "$1" in
        start)
                start
                ;;
        stop)
                stop
                ;;
        restart)
                restart
```

```
                ;;
        status)
                status
                ;;
        *)
                echo $"Usage: $0 {start|stop|restart|status}"
                exit 1
esac

exit $?
```

The permissions on /etc/rc.d/init.d/dspace should be 755. Once you've created the file, you can check that it works by starting, stopping, querying status etc. E.g.

```
sudo /sbin/service dspace start
```

If it's all working you can get it to start up at boot time using the chkconfig system. First, add it to the chkconfig system.

```
sudo /sbin/chkconfig \--add dspace
```

Then get it to start up automatically

```
sudo /sbin/chkconfig dspace on
```

Set up the Database

If the postgresql server has never been started before, make sure to start it so that all the default configuration files get created.

```
/sbin/service postgresql start
```

Edit */var/lib/pgsql/data/pg_hba.conf* and ensure that this is the first authentication line that is uncommented.

```
host  dspace  dspace  127.0.0.1/32  trust
```

N.B. this is good for testing, but tighten up the method for production (e.g. replace "trust" with "md5")

Set PostGreSQL server to start up when the server boots.

```
/sbin/chkconfig postgresql on
```

If you haven't already started postgresql in a previous step, then start the server.

```
/sbin/service postgresql start
```

Create the dspace postgresql user (need to be postgres user):

```
# su -l postgres
$ createuser -U postgres -d -A -P dspace
Enter password for new role:
Enter it again:
Shall the new role be allowed to create more new roles? (y/n) n
CREATE ROLE
$
```

Create the dspace database (need to be dspace unix user):

```
# su -l dspace
$
$ createdb -U dspace -E UNICODE -T template0 dspace
CREATE DATABASE
$
```

Build and Install DSpace

(See also the detailed instructions.)

Download the latest (stable) release of DSpace and put it in **/home/dspace/**.

Most of the following steps should be done as the **dspace** user. Keep an eye out for exceptions.

Extract the tarball (e.g. if you got the bzip2ed version):

```
tar \-jxvf dspace-1.4.2-source.tbz2
```

Copy the PostGreSQL jdbc into the source tree:

```
cp /usr/share/java/postgresql-jdbc3.jar dspace-1.4.2-source/lib/postgresql.jar
```

Edit the **dspace.cfg** (in the config directory of the source tree). Things you probably should customise:

- dspace.dir
- dspace.url
- dspace.hostname
- dspace.name
- db.password
- mail.server
- mail.from.address
- feedback.recipient
- mail.admin
- alert.recipient

Make sure that the directory defined in *dspace.dir* exists and is owned by and writable by the dspace user. The same goes for any other directory such as *assetstore.dir* if you customised that. For this example, we define our dspace.dir as **/data/dspace**, so as **root** we create this directory and change ownership:

```
mkdir -p /data/dspace
chown dspace:dspace /data/dspace
```

Now, back as the dspace user, change to the source directory

```
cd dspace-1.4.2-source
```

Build the war files

```
ant fresh_install
```

you will see a whole bunch of output, which should finish off with some instructions as well as something like this:

```
BUILD SUCCESSFUL
Total time: 28 seconds
```

Copy the resultant war files to the tomcat webapps directory

```
cp \-Rpv build/*.war /usr/local/tomcat/webapps/
```

Create the initial DSpace administrator

```
/data/dspace/bin/create-administrator
```

Make DSpace the root webapp, by symlinking ROOT to the *dspace* directory. N.B. the *dspace* directory will only be created from the dspace.war file the first time tomcat is started, but Unix lets you create symlinks to files that don't exist. As root:

```
ln -s /usr/local/tomcat/webapps/dspace /usr/local/tomcat/webapps/ROOT
```

Start up dspace. As root:

```
/sbin/service dspace start
```

You should now be able to see DSpace running as the root application on port 8080 of your host, e.g. http://www.example.com:8080/

Set Up mod_proxy_ajp

You've got tomcat running, by default it will also be listening on the AJP port, so you can use Apache httpd to act as the point of entry for your site.

Edit /etc/httpd/conf.d/proxy_ajp.conf and add this to it:

```
ProxyPass               /               ajp://localhost:8009/
```

Start up httpd

```
/sbin/service httpd start
```

Check your site, e.g. go to http://www.example.com/ with your browser.

Set up httpd to start up automatically when the server boots

```
/sbin/chkconfig httpd on
```

Further Documentation To Do

- Try OpenJDK from Fedora Linux EPEL repository to replace Sun Java 1.5.

- directory structure suggestions
- security improvements
    - PostGreSQL
    - File ownership
    - SELinux