



## ***CHARMe node Installation Document***



CHARMe is funded by the EC under its FP7 Research Programme

## Document Control

### Contributors

Person	Role	Organisation	Contribution
A.Wilson	Developer	STFC	Initial Draft.

### Document Approval

Person	Role	Organisation

### References

ID	Author	Document Title	Date
[R-1]			

### Revision History

Issue	Author	Date	Description
0.1	A.Wilson	22 <sup>nd</sup> Jan 2015	Initial Draft.
0.2	A.Wilson	17 <sup>th</sup> Feb 2015	Added Strabon instructions

## Table of Contents

1 About these Instructions.....	4
2 Dependencies.....	4
3 Web Server.....	5
3.1 Certificate.....	5
3.2 Security.....	5
3.3 Additional Configuration for Fuseki.....	5
3.4 Start Up.....	5
4 Triple Store.....	6
4.1 Fuseki.....	6
4.1.1 Installation.....	6
4.1.2 Configuration.....	6
5 CHARMe.....	9
5.1 Installation.....	9
5.2 Configuration.....	9
5.2.1 local_settings.py.....	9
5.2.2 wsgi.py.....	10
5.2.3 djcharme_wsgi.conf.....	10
5.2.4 Other Configuration.....	10
5.2.5 Set up the Client via the GUI.....	11
5.2.6 oauth_test2.html.....	11
6 Strabon.....	12

# Node Services

The CHARMe node is a Python Django application that sits behind an Apache web server.

## 1 About these Instructions

The instructions in this document have been tested on Redhat Enterprise Linux and relate to the installation of the central CHARMe node at STFC. When installing on a different node the text in **green** will need to be changed. The text in **red** WILL need to be changed for all installations.

## 2 Dependencies

The server has dependencies on java, python-virtualenv, gcc, mod\_ssl and mod\_wsgi. These can be installed with:

```
yum install java-1.7.0-openjdk.x86_64 python-virtualenv.noarch gcc mod_ssl mod_wsgi
```

It is possible to use one of a number of databases for storing the admin data. For a production node it is recommended that postgres is used.

## 3 Web Server

An Apache web server is used as a front end for Fuseki and CHARMe.

### 3.1 Certificate

Apache should be configured to use a certificate, such as a Comodo certificate.

### 3.2 Security

There are a number of things to do to tighten security. Turn off TRACE, in **httpd.conf** add the following:

```
TraceEnable off
```

Disable SSL v2 and low level ciphers, in **ssl.conf**:

```
SSLProtocol all -SSLv2 -SSLv3
SSLCipherSuite ALL:!ADH:!EXPORT:!LOW:!SSLv2:!RC4:+RSA:+HIGH:+MEDIUM
```

### 3.3 Additional Configuration For Fuseki

In order to give access to the Fuseki query page and end point, add the following to **httpd.conf**:

```
ProxyPass /sparql.html http://127.0.0.1:3333/sparql.html
ProxyPassReverse /sparql.html http://127.0.0.1:3333/sparql.html
ProxyPass /sparql http://127.0.0.1:3333/privateds/sparql
ProxyPassReverse /sparql http://127.0.0.1:3333/privateds/sparql
ProxyPass /fuseki.css http://127.0.0.1:3333/fuseki.css
ProxyPassReverse /fuseki.css http://127.0.0.1:3333/fuseki.css
```

### 3.4 Start Up

Start the service:

```
/etc/init.d/httpd start
chkconfig httpd on
```

## 4 Triple Store

The CHARMe node has been tested with Fuseki and Strabon.

### 4.1 Fuseki

#### 4.1.1 Installation

Create the required directories:

```
mkdir -p /opt/charme/luceneDB /var/log/fuseki
chown apache /opt/charme
```

Get the latest version of Fuseki and unpack it:

```
cd /opt/
wget http://mirror.vorboss.net/apache/jena/binaries/jena-fuseki-1.1.1-
distribution.zip
unzip jena-fuseki-1.1.1-distribution.zip
mv jena-fuseki-1.1.1 jena-fuseki
```

#### 4.1.2 Configuration

Set up the start up script:

```
cp /opt/jena-fuseki/fuseki /etc/init.d/
```

In **/etc/init.d/fuseki** add the FUSEKI\_\* values and edit JAVA\_OPTIONS:

```
export FUSEKI_HOME="/opt/jena-fuseki"
export FUSEKI_ARGS="--update --port=3333 --config=/opt/charme/config-charme.ttl"
export FUSEKI_DATA_DIR="/opt/charme/DB"
export FUSEKI_LOGS="/var/log/fuseki"
JAVA_OPTIONS="-Dlog4j.configuration=file:/opt/jena-fuseki/log4j.properties
-Xmx1200M"
```

Create the file **/opt/charme/config-charme.ttl** with the contents:

```

# Licensed under the terms of http://www.apache.org/licenses/LICENSE-2.0

## Example of a TDB dataset published using Fuseki: persistent storage.

@prefix :      <#> .
@prefix fuseki: <http://jena.apache.org/fuseki#> .
@prefix rdf:   <http://www.w3.org/1999/02/22-rdf-syntax-ns#> .
@prefix rdfs:  <http://www.w3.org/2000/01/rdf-schema#> .
@prefix tdb:   <http://jena.hpl.hp.com/2008/tdb#> .
@prefix ja:    <http://jena.hpl.hp.com/2005/11/Assembler#> .
@prefix text:  <http://jena.apache.org/text#> .
@prefix dcterms: <http://purl.org/dc/terms/> .
@prefix cnt:   <http://www.w3.org/2011/content#>

[] rdf:type fuseki:Server ;
  # Timeout - server-wide default: milliseconds.
  # Format 1: "1000" -- 1 second timeout
  # Format 2: "10000,60000" -- 10s timeout to first result, then 60s timeout to for
rest of query.
  # See java doc for ARQ.queryTimeout
  # ja:context [ ja:cxtName "arq:queryTimeout" ; ja:cxtValue "10000" ] ;
  # ja:loadClass "your.code.Class" ;

  fuseki:services (
    <#service_tdb_all>
    <#service_text_tdb>
  ) .

# TDB
[] ja:loadClass "com.hp.hpl.jena.tdb.TDB" .
tdb:DatasetTDB rdfs:subClassOf ja:RDFDataset .
tdb:GraphTDB rdfs:subClassOf ja:Model .

## Initialize text query
[] ja:loadClass "org.apache.jena.query.text.TextQuery" .
# A TextDataset is a regular dataset with a text index.
text:TextDataset rdfs:subClassOf ja:RDFDataset .
# Lucene index
text:TextIndexLucene rdfs:subClassOf text:TextIndex .

# Text index description
<#indexLucene> a text:TextIndexLucene ;
  text:directory <file:/opt/charme/luceneDB> ;
  text:entityMap <#entMap> ;
  .

# Mapping in the index
# URI stored in field "uri"
# rdfs:label is mapped to field "text"
<#entMap> a text:EntityMap ;
  text:entityField "uri" ;
  text:defaultField "title" ;
  text:field "creator" ;
  text:field "comment" ;
  text:map (
    [ text:field "title" ; text:predicate dcterms:title ]
    [ text:field "creator" ; text:predicate dcterms:creator ]
    [ text:field "comment" ; text:predicate cnt:chars ]
  ) .

```



```

## -----
### This URI must be fixed - it's used to assemble the text dataset.
#
:text_dataset rdf:type      text:TextDataset ;
    text:dataset    <#tdb_dataset_readwrite> ;
    text:index      <#indexLucene> ;
    .

<#service_text_tdb> rdf:type fuseki:Service ;
    rdfs:label      "TDB Service (full_text)" ;
    fuseki:name      "privateds" ;
    fuseki:serviceQuery      "query" ;
    fuseki:serviceQuery      "sparql" ;
    fuseki:serviceUpdate      "update" ;
    fuseki:serviceUpload      "upload" ;
    fuseki:serviceReadWriteGraphStore      "data" ;
    # A separate read-only graph store endpoint:
    fuseki:serviceReadGraphStore      "get" ;
    fuseki:dataset      :text_dataset ;
    .

## -----
## Updatable TDB dataset with all services enabled.

<#service_tdb_all> rdf:type fuseki:Service ;
    rdfs:label      "TDB Service (RW)" ;
    fuseki:name      "privateds" ;
    fuseki:serviceQuery      "query" ;
    fuseki:serviceQuery      "sparql" ;
    fuseki:serviceUpdate      "update" ;
    fuseki:serviceUpload      "upload" ;
    fuseki:serviceReadWriteGraphStore      "data" ;
    # A separate read-only graph store endpoint:
    fuseki:serviceReadGraphStore      "get" ;
    fuseki:dataset      <#tdb_dataset_readwrite> ;
    .

<#tdb_dataset_readwrite> rdf:type      tdb:DatasetTDB ;
    tdb:location      "/opt/charme/DB" ;
    tdb:unionDefaultGraph true ;
##      # Query timeout on this dataset (milliseconds)
##      ja:context [ ja:cxtName "arq:queryTimeout" ; ja:cxtValue "1000" ] ;
##      # Default graph for query is the (read-only) union of all named graphs.

```

Start the services:

```

/etc/init.d/fuseki start
chkconfig fuseki on
chown apache:apache /opt/charme/charme.db
/etc/init.d/httpd restart

```

## 5 CHARMe

### 5.1 Installation

Set up the python virtual environment:

```
cd /opt/  
virtualenv djcharme  
source djcharme/bin/activate  
cd djcharme/  
export http_proxy=http://wwwcache.rl.ac.uk:8080  
export https_proxy=http://wwwcache.rl.ac.uk:8080  
mkdir /var/www/html/djcharme  
export DJANGO_PROJECT_STATIC_FILES=/var/www/html/djcharme/
```

Set the version of the code you wish to install:

```
export VERSION=0.7.2
```

Get the code:

```
wget https://github.com/cedadev/djcharme/raw/develop/djcharme/dist/djcharme-${VERSION}.tar.gz
```

Install the postgres python libraries and the CHARMe code:

```
pip install psycopg2  
pip install djcharme-${VERSION}.tar.gz --extra-index-url http://dist.ceda.ac.uk/pip/
```

### 5.2 Configuration

#### 5.2.1 local\_settings.py

This shows the configuration required for using postgres as the admin database. Edit **/opt/djcharme/lib/python2.6/site-packages/djcharme/local\_settings.py**.

In the DATABASES section set values for ENGINE, NAME, USER and PASSWORD, e.g.

```
'ENGINE': 'django.db.backends.postgresql_psycopg2',  
'NAME': 'charmedb',  
'USER': 'postgresUser',  
'PASSWORD': 'myPassword',
```

Also set values for `STATIC_ROOT` and `NODE_URI`, e.g.

```
STATIC_ROOT = '/var/www/html/djcharme/'  
NODE_URI = 'https://charme.cems.rl.ac.uk'
```

If needed, set the values for `HTTP_PROXY` and `HTTP_PROXY_PORT`.

Update the email settings as required. `DEFAULT_FROM_EMAIL` will need to be set, e.g.

```
DEFAULT_FROM_EMAIL = 'no-reply@charme.cems.rl.ac.uk'
```

### 5.2.2 wsgi.py

Update **wsgi.py**:

```
sed -i s+VEPATH_PAR+/opt/djcharme/lib/python2.6/site-packages+  
/opt/djcharme/lib/python2.6/site-packages/djcharme/resources/wsgi.py  
sed -i s+DJANGO_PATH+/opt/djcharme/lib/python2.6/site-packages/djcharme+  
/opt/djcharme/lib/python2.6/site-packages/djcharme/resources/wsgi.py  
sed -i s+PROJECT_LIB_PATH+/opt/djcharme/lib/python2.6/site-packages/djcharme+  
/opt/djcharme/lib/python2.6/site-packages/djcharme/resources/wsgi.py
```

### 5.2.3 djcharme\_wsgi.conf

Set up the **djcharme\_wsgi.conf**:

```
cp /opt/djcharme/lib/python2.6/site-  
packages/djcharme/resources/djcharme_wsgi.conf /etc/httpd/conf.d/  
sed -i s+LOG_DIR_PATH_PAR/PROJECT_NAME_PAR+/etc/httpd/logs/djcharme+  
/etc/httpd/conf.d/djcharme_wsgi.conf  
sed -i s+PATH_TO_DJANGO_PROJECT_STATIC_FILES+/var/www/html/djcharme/+  
/etc/httpd/conf.d/djcharme_wsgi.conf  
sed -i s+PATH_TO_PROJECT_WSGI+/opt/djcharme/lib/python2.6/site-  
packages/djcharme/resources+ /etc/httpd/conf.d/djcharme_wsgi.conf  
sed -i s+PROJECT_NAME_PAR++ /etc/httpd/conf.d/djcharme_wsgi.conf
```

### 5.2.4 Other Configuration

Initialise the database:

```
python /opt/djcharme/lib/python2.6/site-packages/djcharme/manage.py collectstatic  
--clear --noinput  
python /opt/djcharme/lib/python2.6/site-packages/djcharme/manage.py syncdb --noinput
```

Set up the admin user:

```
python /opt/djcharme/lib/python2.6/site-packages/djcharme/manage.py createsuperuser
```

Deactivate the python environment:

```
deactivate
```

Restart the service:

```
/etc/init.d/httpd restart
```

### 5.2.5 Set up the Client via the GUI

In a browser go to the admin interface:

<https://charme.cems.rl.ac.uk/admin/>

Then go to clients, Add client and provide the relevant values, e.g.

Url: <https://charme.cems.rl.ac.uk/>

Redirect uri: <https://charme.cems.rl.ac.uk/>

Client type: Public (Native and JS applications)

Organization: STFC

Make a note of the Client id.

### 5.2.6 oauth\_test2.html

Edit /opt/djcharme/lib/python2.6/site-packages/djcharme/templates/oauth\_test2.html.

Set the values of oa\_domain and client\_id, using the client id from above, e.g.

```
oa_domain='https://charme-test.cems.rl.ac.uk'  
client_id='1234567890'
```

Restart the service:

```
/etc/init.d/httpd restart
```

## 6 Strabon

Only very basic testing has been undertaken with the Strabon triple store. To use Strabon in place of Fuseki a number of changes are needed in the file `local_settings.py`, i.e.:

```
SPARQL_PORT = "8080"
SPARQL_DATASET = "strabonendpoint"
SPARQL_UPDATE = _format_sparql_url("Update")
SPARQL_QUERY = _format_sparql_url("Query")
```

In addition to these changes three extra properties are required:

```
SPARQL_USERNAME = "endpoint"
SPARQL_PASSWORD = "myPassword"
STRABON = True
```

This should be all of the extra configuration required to run the CHARMe node using Strabon.