MANUAL DE INSTALACIÓN

Versión: 2.0.0/24-ene-2012

1. Introducción

1.1 Objeto

En este documento se describe la instalación simplificada de los componentes necesarios y de la aplicación OpenIrekia en un servidor Ubuntu Linux 8.04 LTS. La instalación creada tiene todo los componentes necesarios para y desarrollo con el sistema.

1.2 Pre-requisitos

Para poder seguir el manual es necesario disponer de:

- Servidor con sistema operativo Ubuntu Server 8.04 LTS ya instalado y actualizado y acceso como root
- Conexión a Internet

1.3 Público

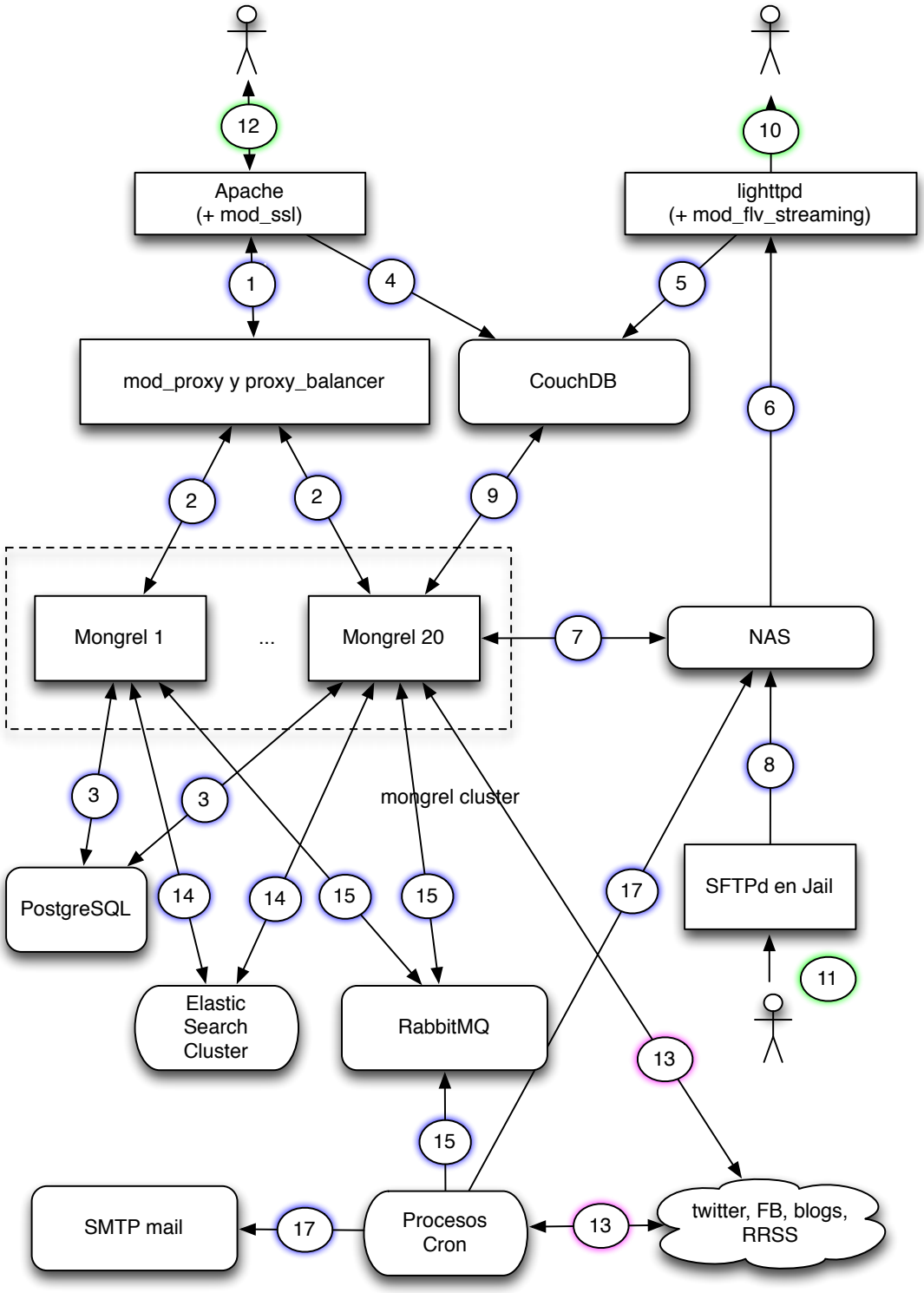
Para realizar la instalación es necesario tener conocimientos de administración de un sistema Linux/Unix.

1.4. Licencia

El Gobierno Vasco pone a disposición de usuarios, desarrolladores y comunidad en general la aplicación denominada "OpenIrekia – Gobierno Abierto" bajo la Licencia Pública de la Unión Europea "European Union Public Licence – EUPL". Esta licencia, desarrollada en el seno de la Unión Europea, nació con la intención de ser la licencia bajo la cuál se liberasen los programas y aplicaciones desarrolladas por la Administración Pública y con la característica específica de ser compatible con otras licencias denominadas libres, como la GNU General Public License (GNU/GPL). Estas características dotan, a las aplicaciones así liberadas, de mayor seguridad jurídica y fomentan la interoperabilidad de los servicios de la Administración Electrónica.

2. Estructura y componentes

En la siguiente figura se muestran los principales componentes del sistema en un instalación típica. En una instalación de desarrollo no sería necesario usar NAS externo para el contenido multimedia, cluster de mongrels, y acceso restringido para colaboradores por SFTP en jail.



1. El servidor Apache con VirtualHost hasta cluster de 20 copias de Mongrel.
   Ficheros de configuración relevantes: /etc/httpd/conf/httpd.conf y /etc/httpd/conf.d/ssl.conf
   Los procesos se ejecutan con permisos del usuario apache.

2. El proxy_balancer reparte las peticiones entrantes entre el cluster de Mongrels
   Ficheros de configuración relevantes: /etc/httpd/conf.d/proxy_balancer.conf y /etc/mongrel_cluster/ogov.yml
   Protocolo de conexión: HTTP puertos 8080-8099
   Los procesos se ejecutan con permisos del usuario apache.

3. La aplicación usa conexión vía socket local hasta la base de datos postgres
   Ficheros de configuración relevantes: /usr/app/ogov/config/database.yml
   Protocolo de conexión: socket puerto 5432
   Los procesos se ejecutan con permisos de usuarios ogov y postgres.

4. Los logs de acceso por HTTP y HTTPS al Apache se registran en la base de datos CouchDB
   Ficheros de configuración relevantes: /etc/httpd/conf/httpd.conf /etc/httpd/conf.d/ssl.conf y
   /usr/local/etc/couchdb/default.ini
   Protocolo de conexión: REST vía HTTP a puerto 8984
   Los procesos se ejecutan con permisos del usuario apache y couchdb.

5. Los logs de acceso por HTTP para el contenido multimedia (incluido el streaming vía mod_flv_streaming) se registran en la base de datos CouchDB.
   Ficheros de configuración relevantes: /etc/lighttpd/lighttpd.conf y /usr/local/etc/couchdb/default.ini
   Protocolo de conexión: REST vía HTTP a puerto 8984

6. El lighttpd usa el ficheros multimedia desde el NAS en /web
   Ficheros de configuración relevantes: /etc/lighttpd/lighttpd.conf y /etc/fstab
   Protocolo de conexión: NFS

7. Las aplicaciones crean y trasladan los directorios necesarios y preparan los vídeos en .flv para streaming
   Ficheros de configuración relevantes: /etc/fstab
   Protocolo de conexión: NFS

8. El servidor de SFTP permite deposita el contenido multimedia en el NAS
   Ficheros de configuración relevantes: /etc/fstab y /web/jails/agencia/etc/jailkit/jk_lsh.ini
   Protocolo de conexión: NFS

9. El módulo de estadística en tiempo real accede a las bases de datos en CouchDB
   Ficheros de configuración relevantes: /usr/local/etc/couchdb/default.ini
   Protocolo de conexión: REST vía HTTP a puerto 8984

10. Los usuarios finales acceden a los contenidos multimedia vía navegador web o reproductor flash
    Ficheros de configuración relevantes: /etc/lighttpd/lighttpd.conf
    Protocolo de conexión: HTTP

11. Los colaboradores suben contenido pesado (video y fotos) vía SFTP

12. Los usuarios finales y los internos usuarios autorizados acceden a los servicios web
    Protocolos de conexión: HTTP y HTTPS

13. Las aplicaciones usan servicios externos
    Protocolos de conexión: HTTP y HTTPS

14. Las aplicaciones mantienen el índice de búsqueda el texto completo en el servidor ElasticSearch
    Ficheros de configuración relevantes: /usr/local/elasticsearch/config/elasticsearch.yml
    Protocolo de conexión: HTTP al puerto 9200

15. Mensajes hasta las colas de tareas programadas, conexiones por el puerto 5672

16. Las tareas programadas relacionadas con gestión y transformación de vídeo acceden al NAS por NFS

17. Las aplicaciones (no mostrado) y los procesos periódicos usan SMTP para el envío de alertas por email.

Nota: OpenIrekia gestiona los procesos de streaming en directo pero no incluye servidor de streaming en directo. Actualmente se usa un servicio por parte de los servicios informáticos del gobierno basado en Wowza Media Server y protocolo RTMP.

3. Preparación del servidor

Una vez instalado el sistema operativo básico confirmar la conexión a Internet y seguir los siguientes pasos. Durante la instalación cuando sea posible se usan paquetes de la distribución. Cuando algún componente de software no esta disponible o no esta en la versión necesaria se compila e instala a mano.

3.1 Paquetes de Ubuntu

Instalar los siguientes paquetes de la distribución:

```
sudo apt-get -y install build-essential
sudo apt-get -y install ruby ruby1.8-dev irb rdoc ri libopenssl-ruby1.8
sudo apt-get -y install postgresql-8.3 postgresql-server-dev-8.3
sudo apt-get -y install apache2
sudo apt-get -y install imagemagick libmagick9-dev
sudo apt-get install lighttpd
```

Permitir conexiones locales al servidor PostgreSQL:

```
sudo sed -e \
       's|127.0.0.1/32          md5|127.0.0.1/32          trust|' \
       -i /etc/postgresql/8.3/main/pg_hba.conf
sudo /etc/init.d/postgresql-8.3 restart
```

3.2 Componentes vía RubyGems

Descargar e instalar RubyGems:

```
wget http://production.cf.rubygems.org/rubygems/rubygems-1.3.7.tgz
tar zxvf rubygems-1.3.7.tgz
cd rubygems-1.3.7
sudo ruby setup.rb
sudo ln -s /usr/bin/gem1.8 /usr/bin/gem
```

Instalar los siguientes gems:

```
sudo gem install postgres
sudo gem install mongrel mongrel_cluster
sudo gem install -v=2.2.3 rails
sudo gem install -v=1.0.6 flvtool2
sudo gem install -v=1.4.1 geokit
sudo gem install -v=0.8.5 ri_cal
sudo gem install -v=0.7.9 twitter
sudo gem install uuidtools
sudo gem install -v=1.0.8 afer_commit
```

3.3 Componentes por compilar a mano

3.3.1 flvmeta

Descargar e instalar flvmeta:

```
wget http://www.efaber.net/ogov/flvmeta-1.0.9.tar.gz
tar --no-same-permissions -zxvof flvmeta-1.0.9.tar.gz
cd flvmeta-1.0.9
./configure
make
sudo make install
```

3.3.2 ffmpeg

Descargar e instalar el ffmpeg y sus dependencias:

```
wget http://www.efaber.net/ogov/lame-3.98.4.tar.gz
tar --no-same-permissions -zxvof lame-3.98.4.tar.gz
cd lame-3.98.4
./configure
make && sudo make install

wget http://www.efaber.net/ogov/faac-1.28.tar.gz
tar --no-same-permissions -zxvof faac-1.28.tar.gz
cd faac-1.28
./configure
make && sudo make install

wget http://www.efaber.net/ogov/faad2-2.7.tar.gz
tar --no-same-permissions -zxvof  faad2-2.7.tar.gz
cd faad2-2.7
./configure
make && sudo make install

wget http://www.efaber.net/ogov/x264-snapshot-20100420-2245.tar.bz2
tar -xjvf x264-snapshot-20100420-2245.tar.bz2
cd x264-snapshot-20100420-2245
./configure --enable-shared --disable-asm
make && sudo make install

wget http://www.efaber.net/ogov/ffmpeg-co-20100421.tar.gz
tar --no-same-permissions -zxvof ffmpeg-co-20100421.tar.gz
cd ffmpeg
./configure --enable-gpl --enable-nonfree \
            --enable-pthreads --enable-libfaac \
            --enable-libfaad --enable-libmp3lame --enable-libx264
make && sudo make install

sudo ldconfig
```

3.3.4 stream segmenter

Descargar e instalar el stream segmenter:

```
wget http://www.efaber.net/ogov/segmenter.tar.gz
tar --no-same-permissions -zxvof segmenter.tar.gz
cd segmenter/
```

```
make && sudo make install
```

Añadir los headers correctos para el streaming por http en la configuración del servidor Apache en/etc/apache2/mods-available/mime.conf

```
AddType application/x-mpegURL .m3u8
AddType video/MP2T .ts
```

Añadir los headers correctos para el streaming por http en la configuración del servidor lighttpd en /etc/lighttpd/ o revisando /etc/mime.types

```
".m3u8"  =>  "application/application/x-mpegURL",
".ts"    =>  "video/MP2T",
```

3.3.5 generador QR

Descargar e instalar el generador de códigos QR y la librería libpng:

```
sudo apt-get install libpng-dev
apt-get install pkg-config
cd /usr/local/src/
wget http://www.efaber.net/ogov/qrencode-3.1.1.tar.gz
tar --no-same-permissions -zxvof qrencode-3.1.1.tar.gz
cd qrencode-3.1.1/
./configure
make
make install
```

3.5 Proxy de Apache

Crear proxy_balancer.conf con contenido del tipo:

```
<Proxy balancer://ogovcluster>
  BalancerMember http://-ip-interna-:8080
  BalancerMember http://-ip-interna-:8081
  BalancerMember http://-ip-interna-:8082
  ...
</Proxy>
```

Configurar el acceso por HTTPS en ssl.conf prestando atención a los siguientes parámetros:

```
<VirtualHost -ip-interna-:443>
  ServerName -nombre-publico-
  SSLEngine on
  SSLProtocol all -SSLv2
  SSLCipherSuite ALL:!ADH:!EXPORT:!SSLv2:RC4+RSA:+HIGH:+MEDIUM:+LOW
  SSLCertificateFile -path-al.cer-
  SSLCertificateKeyFile -path-al.key-

  SetEnvIf User-Agent ".*MSIE.*" \
         nokeepalive ssl-unclean-shutdown \
         downgrade-1.0 force-response-1.0

  ProxyRequests Off
  ProxyPass / balancer://ogovcluster/
  ProxyPassReverse / balancer://ogovcluster/
  RequestHeader set X_FORWARDED_PROTO "https"
```

```
</VirtualHost>
```

Finalmente configurar el VirtualHost para el numbre público con atención a los siguientes parámetros:

```
<VirtualHost -ip-:80>
ServerName -nombre-publico-

RewriteEngine On
RewriteCond %{HTTPS} !=on
RewriteRule ^/(..)/ma/session/new https://%{SERVER_NAME}/$1/ma/session/new [R,L]
RewriteRule ^/sadmin(.*) https://%{SERVER_NAME}/sadmin$1 [R,L]
RewriteRule ^/admin(.*) https://%{SERVER_NAME}/admin$1 [R,L]

ProxyRequests Off
ProxyPass / balancer://ogovcluster/
ProxyPassReverse / balancer://ogovcluster/
...
</VirtualHost>
```

3.8 Componentes para estadística en tiempo real

Para el módulo de estadística en tiempo real instalar los pre-requisitos para Erlang:

```
sudo apt-get -y install libcurl4-openssl-dev libssl-dev
sudo apt-get -y install libssh2-1-dev openssl
sudo apt-get -y install libreadline5-dev checkinstall libmozjs-dev
sudo apt-get -y install libicu38 libicu-dev curl
```

Instalar Erlang:

```
cd /usr/local/src/
wget http://www.erlang.org/download/otp_src_R14B_erts-5.8.1.1.tar.gz
tar --no-same-permissions -ozxvf otp_src_R14B_erts-5.8.1.1.tar.gz
cd otp_src_R14B
./configure --enable-threads --enable-smp-support
make && sudo make install
```

Instalar CouchDB:

```
wget http://apache.rediris.es/couchdb/1.0.1/apache-couchdb-1.0.1.tar.gz
tar zxvf apache-couchdb-1.0.1.tar.gz
cd apache-couchdb-1.0.1/
./configure
make && sudo make install

sudo useradd -r --comment "CouchDB Owner" couchdb
sudo chown couchdb:couchdb /usr/local/var/log/couchdb
sudo chown couchdb:couchdb /usr/local/var/lib/couchdb
sudo ln -s /usr/local/etc/init.d/couchdb /etc/init.d/couchdb
```

Iniciar el CouchDB con /etc/init.d/couchdb start y comprobar que funciona corectamente vía Futon.

Activar el logging desde los servidores Apache y lighttpd al CouchDB:

log-reader:

```
sudo apt-get install python-dev python-simplejson python-httplib2
wget http://www.efaber.net/ogov/log_reader_git.tar.gz
tar --no-same-permissions -oxzvf log_reader.tar_git.gz
cd log_reader_git
sudo python setup.py install
```

python-couchdb:

```
wget http://www.efaber.net/ogov/CouchDB-0.6.tar.gz
tar --no-same-permissions -ozxvf CouchDB-0.6.tar.gz
cd CouchDB-0.6
sudo python setup.py install
```

Crear la base de datos para logs:

```
curl -X PUT http://localhost:5984/ilog2
```

Configurar el logging en el Apache, vía /etc/httpd/conf/httpd.conf y /etc/httpd/conf.d/ssl.conf añadiendo en las secciones VirtualHost:

```
CustomLog "|python /usr/local/bin/a2c.py irekia" combined
```

y

```
CustomLog "|python /usr/local/bin/a2c.py irekia_https" combined
```

Configurar el lighttpd en /etc/lighttpd/lighttpd.conf:

```
accesslog.filename = "|python /usr/local/bin/a2c.py video"
```

Finalmente instalar:

```
sudo gem install couchrest
sudo gem install json
```

3.9 Gestor de colas

Instalar el servidor RabbitMQ:

```
sudo apt-get install rabbitmq-server
/etc/init.d/rabbitmq-server start
rabbitmq-plugins enable rabbitmq_management
/etc/init.d/rabbitmq-server stop
/etc/init.d/rabbitmq-server start
rabbitmqctl add_user ogov ogrq12
rabbitmqctl set_permissions ogov ".*" ".*" ".*"
```

Si es necesario comprobar la configuración vía HTTP al puerto 55672

En todos los servidores instalar el cliente:

```
gem install -v=0.7.6 bunny
```

3.9 Servidor ElasticSearch

Instalar ES de :

```
    https://github.com/downloads/elasticsearch/elasticsearch/elasticsearch-0.17
.6.tar.gz
```

4. OpenIrekia

4.1 Usuarios

```
groupadd -g 95 rails
useradd -m -G rails openirekia
sudo -u postgres createuser --no-superuser \
        --createdb --no-createrole openirekia
```

4.2 Aplicación OpenIrekia

Crear directorio (por ejemplo /srv/openirekia) que pertenezca al usuario openirekia y el grupo rails y descomprimir el openirekia-1.0.0.tar.gz

```
cd /srv/openirekia
tar -zxvf OpenIrekia-2.0.0.tgz
```

Para incluir contenidos multimedia de ejemplo:

```
cd openirekia/public/mm
tar -zxvf OpenIrekia-mmedia-1.0.0.tgz
```

Crear los ficheros database.yml y environment.rb en el directorio config. Se pueden usar como ejemplo los ficheros database-openirekia.yml e environment-openirekia.rb.

Llamar al fichero de configuración de los directorios para los contenidos multimedia:

```
require "#{RAILS_ROOT}/config/multimedia_directories-openirekia.rb"
```

Cambiar los valores relacionados con la sesión:

```
config.action_controller.session = {
   :session_key => '_open_irekia_session',
   :secret      => 'YOUR_SECRET_KEY'
}
```

Crear la base de datos inicial:

```
rake db:create:all
rake db:schema:load
rake openirekia:load
rake db:migrate
rake search:index_items_to_elasticsearch
rake search:index_items_to_ma_elasticsearch
rake external:news_to_elastic_search
```

Después de ejecutar rake openirekia:load la base de datos contiene varios ejemplos de diferentes tipos de contenido y varios usuarios:

ROLE	EMAIL	CONTRASEÑA
Administrador	admin@example.com	openirekia
Jefe de departamento	jefe_dept@example.com	openirekia
Periodista	periodista@example.com	openirekia
Operador de streaming	operador@example.com	openirekia
Usuario registrado	usuario@example.com	openirekia

4.3 Otros parámetros de configuración

- initializers/geokit_config.rb : Fichero de configuración de Geokit, la librería para geolocalizar direcciones. Hay que sustituir el valor del "Google Maps geocoder key". Por defecto se usa el de localhost
- initializers/flowplayer.rb : Configuración del flowplayer.
- config/bitly.yml: Configuración para bit.ly
- config/twitter.yml: Configuración para el login a través de Twitter y los twits automáticos de noticias y eventos.

app/models/comment.rb: Configuración para Akismet. Los valores correspondientes están dentro de la función akismet_attributes:

```
def akismet_attributes # :nodoc:
   {
     :key                  => 'YOUR_KEY',
     :blog                 => 'YOUR_WEB',
     ....
   }
 end
```

- initializers/google_api.rb_: Configuración para el tracking con Google Analytics. Poner el API ID que nos da Google.
- config/elastic_search_server.rb

4.4 Tests

Antes de ejecutar los tests hay que configurar los valores para algunas constantes en el fichero config/environments/test.rb

```
DocumentPaths::MULTIMEDIA_PATH = path_to_mm_files_for_test
PhotoPaths::PHOTOS_PATH = path_to_photos_for_test
```

4.5 Procesos periódicos

Para el correcto funcionamiento del sistema se ejecutar periódicamente varios procesos. Los propios programas son scripts de rake y bash, se encuenran en batch_processes y se pueden ejecutar a través de cron.

`process_unsent_alerts_for_journalists.sh` -- Recorre y envia la cola de alertas por enviar a periodistas.

`process_unsent_alerts_for_staff.sh` -- Recorre y envia la cola de alertas por enviar responsables de salas de streaming y operadores.

`include_new_videos_in_webtv.sh` -- Busca videos nuevos en las noticias del último mes, y los incluye en la WebTV.

`include_new_photos_in_gallery.sh` -- Busca fotos nuevas en las noticias del último mes, y los incluye en la fototeca.

`tweet_pending_issues.sh` -- Recorre y envia la cola de tweets sobre nuevas noticias y eventos.

`check_ubervu_updates.sh` -- Comprueba nuevas menciones de URLs.

`add-meta.sh` -- Añade metadatos necesarios para el streaming.

`cdb-ping-views.rb` -- Actualiza los views del CouchDB.

`frame_extractor.rb` -- Extrae fotos candidatas para usar como carátula en los vídeos

`ts_streamer.rb` -- Prepara versiones de los vídeos para clientes iOS

`html5_renew.rb` -- Prepara la versión de los vídeos para iPad