

## MANUAL DE INSTALACIÓN

Versión: 1.0.0/10-dic-2010



1. Introducción	2
1.1 Objeto	2
1.2 Pre-requisitos	2
1.3 Público	2
1.4. Licencia	2
2. Estructura y componentes	3
3. Preparación del servidor	5
3.1 Paquetes de Ubuntu	5
3.2 Componentes vía RubyGems	5
3.3 Componentes por compilar a mano	6
3.3.1 flvmeta	6
3.3.2 ffmpeg	6
3.3.3 stream segmenter	6
3.4 Proxy de Apache	7
3.5 Componentes para estadística en tiempo real	8
4. OpenIrekia	10
4.1 Usuarios	10
4.2 Aplicación OpenIrekia	10
4.3 Parámetros de configuración de servicios externos	11
4.5 Procesos periódicos	11

# 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.

The European Union Public Licence <http://www.osor.eu/eupl>

EUPL v.1.1 - Preámbulo

EUPL v.1.1 – Licencia

Copyright 2009-2010 eFaber, S.L.

Copyright 2009-2010 Ejie, S.A.

Copyright 2009-2010 Dirección de Gobierno Abierto y Comunicación en Internet; Gobernu Irekirako eta Interneteko Komunikaziorako Zuzendaritza; Lehendakaritza. Gobierno Vasco – Eusko Jaurlaritza

Licencia con arreglo a la EUPL, Versión 1.1 o –en cuanto sean aprobadas por la Comisión Europea– versiones posteriores de la EUPL (la Licencia);

Solo podrá usarse esta obra si se respeta la Licencia.

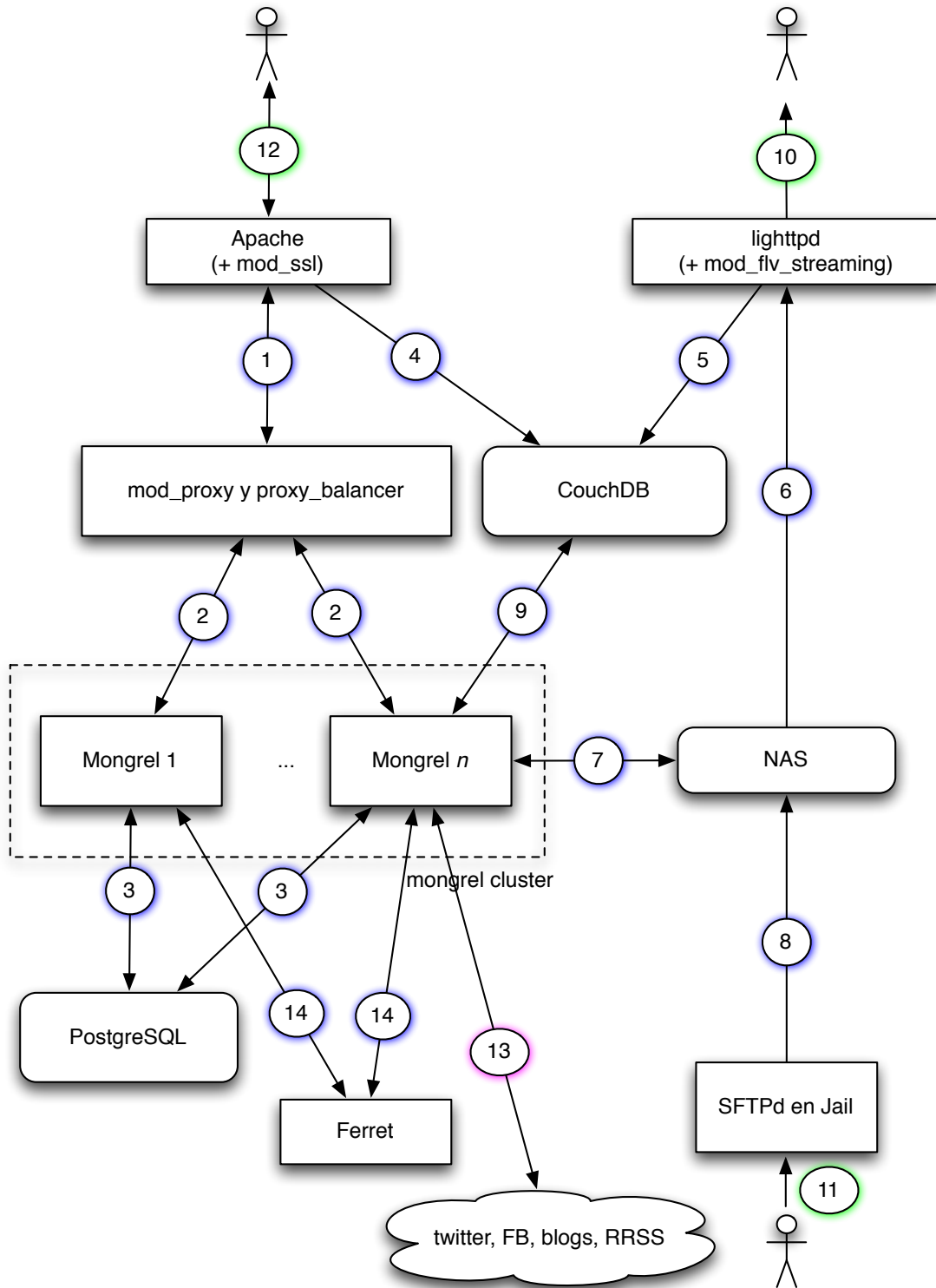
Puede obtenerse una copia de la Licencia en:

<http://ec.europa.eu/idabc/eupl>

Salvo cuando lo exija la legislación aplicable o se acuerde por escrito, el programa distribuido con arreglo a la Licencia se distribuye TAL CUAL, SIN GARANTÍAS NI CONDICIONES DE NINGÚN TIPO, ni expresas ni implícitas. Véase la Licencia en el idioma concreto que rige los permisos y limitaciones que establece la Licencia

## 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  $n$  copias de Mongrel.  
Los procesos se ejecutan con permisos del usuario www-data.
2. El proxy\_balancer reparte las peticiones entrantes entre el cluster de Mongrels  
Protocolo de conexión: HTTP puertos  $80nn-80nn$   
Los procesos se ejecutan con permisos del usuario apache.
3. La aplicación usa conexión vía socket local hasta la base de datos postgres  
Protocolo de conexión: socket puerto 5432  
Los procesos se ejecutan con permisos de usuarios openirekia y postgres.
4. Los logs de acceso por HTTP y HTTPS al Apache se registran en la base de datos CouchDB  
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.  
Protocolo de conexión: REST vía HTTP a puerto 8984
6. El lighttpd usa el ficheros multimedia desde NAS  
Protocolo de conexión: NFS
7. Las aplicaciones crean y trasladan los directorios necesarios y preparan los vídeos en .flv para streaming  
Protocolo de conexión: NFS
8. El servidor de SFTP en jail que permite depositar contenido multimedia en el NAS  
Protocolo de conexión: NFS
9. El módulo de estadística en tiempo real accede a las bases de datos en CouchDB  
Protocolo de conexión: REST vía HTTP a puerto 5984
10. Los usuarios finales acceden a los contenidos multimedia vía navegador web o reproductor flash  
Protocolo de conexión: HTTP
11. Los colaboradores suben contenido pesado (video y fotos) vía SFTP con usuarios en jail
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 Ferret  
Protocolo de conexión: DRb al puerto 9010
15. (no mostrado) Las aplicaciones 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=0.11.6 ferret
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 -v=0.2 rubervu
```

### 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
```

#### 3.3.3 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/

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

### 3.4 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 número 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>
```

### 3.5 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 correctamente 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 -ozxvf 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
```

## 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-1.0.0.tar.gz
```

Si se quieren incluir ficheros multimedia en los datos iniciales que se cargarán en la base de datos

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

Crear los ficheros de configuración database.yml y environment.rb a partir de los ficheros crear los ficheros database-openirekia.yml y environment-openirekia.rb proporcionados. Especialmente se deben cambiar los parámetros relacionados con las cookies

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

Creación de la base de datos y carga de los datos iniciales de prueba:

```
cd /srv/openirekia
rake db:create:all
rake db:schema:load
rake openirekia:load
```

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

Para iniciar el sistema en modo de desarrollo en <http://localhost:3000>:

script/server

Para uso en producción configurar cluster de Mongrels que coincida con los esperados desde el módulo proxy del Apache (punto 3.5).

### 4.3 Parámetros de configuración de servicios externos

- Configuración de **Geokit**, la librería para geolocalizar direcciones:

/srv/openirekia/initializers/geokit\_config.rb: Obtener de Google Maps un API KEY para la URL de nuestra web, y sustituir el valor del parámetro "Google Maps geocoder key"

- Configuración para el servicio **UberVu** de seguimiento de reacciones en las redes sociales (<http://www.ubervu.com/>)

/web/openirekia/initializers/ubervu.rb: Obtener un API KEY para nuestro servicio y sustituir el valor de Rubervu::Ubervu::API\_KEY

- Configuración del **Flowplayer**, el reproductor de videos flv (<http://flowplayer.org/>)

/web/openirekia/initializers/flowplayer.rb. Obtener una API KEY y sustituir el valor del parámetro YOUR\_FLOWPLAYER\_KEY

- Configuración para **bit.ly**, el acortador de URLs que se usa en los tweets de noticias y eventos (<http://bit.ly>)

/web/openirekia/config/bitly.yml: Obtener un API KEY y sustituir el valor del parámetro YOUR\_API\_KEY y el nombre de usuario

- Configuración para el login a través de **Twitter** y de las cuentas para los tweets automáticos de noticias y eventos.

/web/openirekia/config/twitter.yml: Hay tres cuentas. La primera es la que debe configurarse para poder solicitar hacer login. La segunda es la que se usa para tweetear las noticias y la tercera para tweetear los eventos.

- Configuración del filtro antispam **Akismet** (<http://akismet.com/>).

/web/openirekia/app/models/comment.rb: Obtener un API KEY y sustituir los valores del parámetro akismet\_attributes

- Configuración para el tracking con **Google Analytics**.

initializers/google\_api.rb: Obtener un API KEY y sustituir los valores.

### 4.5 Procesos periódicos

Para el correcto funcionamiento del sistema se ejecuta periódicamente varios procesos. Los propios programas son scripts de rake y bash, se encuentran en batch\_processes y se pueden ejecutar a través de cron.

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