MEMORIA

Plug-in de procesado visual (OpenCV) en OpenDomo OS
Máster Universitario en Software libre

Especialidad: Desarrollo de aplicaciones de software libre

Autor: Alexander Herrera Castro

Consultor: Gregorio Robles Martínez

Tutor de Prácticas Externas: Oriol Palenzuela i Rosés

Empresa asociada: OpenDomo Services S.L

Fecha: Enero 9 de 2015

Licencia

Resumen

Este proyecto consiste en el diseño y desarrollo de un plug-in que permita usar el sistema de procesado de imagen OpenCV desde el sistema operativo OpenDomo OS, el proyecto es propuesto por OpenDomo Services S.L, empresa que tiene un convenio de operación educativa con la UOC y que trabaja entre otros en el desarrollo de soluciones con software libre para los campos de la domótica, los automatismos y la eficiencia energética.

Este nuevo plug-in llamado OpenCVODOS, esta enfocado a la búsqueda por parte de OpenDomo Services S.L de robustas y nuevas posibilidades para su sistema operativo OpenDomo OS, ya que OpenCVODOS puede jugar un papel muy relevante dentro de una instalación domótica, facilitando a futuro incorporación de filtros de procesamiento gráfico específicos como reconocimiento facial, detección de movimiento, barreras virtuales, reconocimiento de objetos, visión robótica etc.

El desarrollo del proyecto abarca desde el estudio del funcionamiento de OpenCV, la integración de las librerías existentes de Python y de OpenCV en OpenDomo OS, el diseño y desarrollo del plug-in , la documentación del código fuente del plug-in, hasta la implementación de una batería de pruebas que garantice el buen funcionamiento del plug-in, todo esto basado en lenguajes de script ajustándose a la arquitectura de OpenDomo OS.

Tabla de Contenido

Índice de tablas

Índice de ilustraciones

1. Introducción

OpenDomo OS es un sistema operativo libre basado en GNU/Linux desarrollado por OpenDomo Services S.L, diseñado especialmente para el control de automatismos y su aplicación en entornos domésticos como solución domótica [1].

El proyecto propuesto por OpenDomo Services S.L, consiste en el diseño y desarrollo de un Plug-in que permita usar el sistema de procesado de imagen OpenCV desde OpenDomo OS, abriendo la posibilidad a la futura incorporación de plug-ins específicos como reconocimiento facial, detección de movimiento, etc.

Motivación

La motivación por parte de OpenDomo Services S.L, es darle avance a su sistema de visión *opendomo-vision* que para finales del verano de 2014 se publicó en la versión 2.1 de OpenDomo OS, ésta primera versión estable de *opendomo-vision* permite la gestión de cámaras IP y provee un sistema distribuido de videovigilancia, y se enlazará con OpenCV por medio del nuevo plug-in de procesado visual.

OpenDomo considera que éste nuevo plug-in de procesado visual se convertirá en una de las características más potentes del sistema operativo OpenDomo OS, ya que puede jugar un papel muy relevante dentro de una instalación domótica, facilitando la incorporación de nuevas funciones relacionadas con OpenCV, por ejemplo debería ser posible instalar/activar el reconocimiento facial, la detección de movimientos, etc.

La principal motivación del autor esta enfocada al trabajo con GNU/Linux, herramientas libres y sistemas domóticos, estos últimos permiten el uso de la tecnología en los hogares o viviendas, dándole la capacidad al usuario de gestionar:

- El consumo energético en general de la vivienda.
- La seguridad de la vivienda.
- El bienestar o confort en cuanto a iluminación, control de electrodomésticos y control de otros sistemas.
- La comunicación en general con la vivienda.

Todo esto usando Internet, el autor considera que la combinación de todos estos elementos permitirán a futuro acercar y usar de una manera más inteligente la tecnología por parte del ser humano, buscando optimizar los recursos y comodidades que la misma puede brindar [4].

Estructura de la memoria

Esta memoria consta de diferentes apartados que buscan explicar las etapas en el diseño y concepción del plug-in de procesamiento visual usando OpenCV para OpenDomo OS, a continuación se muestra la estructura de esos apartados:

- **Tecnologías utilizadas:** Muestra de manera general las tecnologías a usar en el desarrollo del proyecto, tanto a nivel de hardware como de software.

- **Implementación:** Permite mostrar los procesos involucrados en el desarrollo del plug-in; desde la adecuación de librerías en OpenDomo OS, el diseño y desarrollo del plug-in, la estructura del plug-in y explicaciones respectivas sobre el funcionamiento y uso de los scripts desarrollados.

- **Pruebas y validación del plug-in:** Muestra una serie de pruebas y los resultados de las mismas, esto permite validar el correcto funcionamiento del plug-in.

- **Conclusiones:** Se muestra de manera general los aportes generados al terminar el desarrollo del proyecto.

- **Seguimiento y cronograma de trabajo:** Este punto explica el cronograma seguido con las diferentes dificultades y soluciones a los mismos.

- **Conocimientos Aplicados Adquiridos en el Máster de Software Libre:** Muestra la relación entre los conocimientos adquiridos en el Máster de Software Libre y los conocimientos que se aplicaron en el desarrollo del plug-in.

- **Conocimientos Adquiridos en el TFM:** Se detallan los conocimientos que se obtuvieron a lo largo del desarrollo e implementación del Trabajo Final de Máster.

- **Trabajos Futuros:** En este punto se describen algunos posibles trabajos a largo plazo que pueden contribuir a desarrollar nuevas funcionalidades y herramientas en torno al plug-in y el procesamiento de imágenes en OpenDomo OS.

2. Objetivos

- Desarrollar un Plug-in que permita usar el sistema de procesado de imagen OpenCV desde OpenDomo OS

  Se creará un Plug-in que permita el uso de OpenCV para el procesamiento de imágenes, estas imágenes serán suministradas por el plug-in de procesamiento visual opendomo-vision de OpenDomo OS, el nuevo plug-in deberá:

  - Usar las cámaras IP configuradas por opendomo-vision.
  - Instalar y usar las librerías de OpenCV.
  - Usar uno o varios lenguajes de scripts en la implementación e incorporación de funcionalidades de OpenCV.

Objetivos específicos

- Estudiar el funcionamiento de OpenCV

  Permitirá conocer el funcionamiento de OpenCV en GNU/Linux y de esta forma definir como se puede integrar a OpenDomo OS.

- Integrar librerías existentes para ser usados desde el Plug-in

  Se deberá buscar como integrar las librerías de OpenCV en OpenDomo OS, de manera tal que el plug-in desarrollado instale los paquetes y las dependencias necesarias para el funcionamiento del mismo, esto desde el repositorio de código de OpenDomo OS.

- Diseñar y desarrollar el Plug-in

  Esta tarea consistirá en crear el código fuente del plug-in y el código fuente los diferentes comandos que serán integrados a OpenDomo OS, de manera tal que el usuario pueda relacionar las cámaras existentes con el plug-in y las herramientas que tengan los mismos, además el plug-in quedará como un servicio más de OpenDomo OS.

- Documentar el código fuente del Plug-in desarrollado

  Consistirá en entregar las diferentes instrucciones de uso y las explicaciones referentes al código fuente desarrollado, lo que permitirá a futuro continuar con el desarrollo de nuevas herramientas en torno al plug-in.

- Desarrollar una batería de pruebas que garantice el buen funcionamiento de la aplicación.

  Este ítem permitirá comprobar el funcionamiento del plug-in en OpenDomo OS, validando diferentes situaciones típicas en la configuración y puesta en marcha del plug-in.

Planificación

La planificación del trabajo a realizar para el desarrollo del proyecto tiene una duración de un semestre, que totalizan en la *UOC* un total de 18 semanas de trabajo, se describe en la tabla 1 el calendario de objetivos con las tareas, fechas y los productos a entregar.

Actividad a desarrollar	Fecha Inicio	Fecha Final	Producto a entregar
Inicio Trabajo Final de Máster	17/09/2014	17/09/2014	Ninguno
Crear plan de trabajo	17/09/2014	29/09/2014	Documento plan de trabajo
Entrega PEC 1 Plan de trabajo	29/09/2014	29/09/2014	Primera Prueba de Evaluación Continua
Estudiar el funcionamiento de OpenDomoOS	17/09/2014	10/10/2014	Documento parcial sobre OpenDomoOS
Estudiar el funcionamiento de OpenCV	17/09/2014	15/10/2014	Documento parcial sobre OpenCV
Integrar librerías existentes para ser usados desde el plugin	17/09/2014	24/10/2014	Adecuación de librerías en OpenDomoOS
Diseñar y desarrollar el plugin	27/10/2014	28/11/2014	Plugin de procesado visual Beta 1
Pruebas y validación de la aplicación	28/11/2014	05/12/2014	Documento de pruebas
Crear documentos necesarios para la entrega final	08/12/2014	19/12/2014	Documentos finales
Crear presentación del trabajo final	22/12/2014	02/01/2015	Vídeo a publicar en herramienta Present@
Entrega PEC 2 Trabajo final	02/01/2015	02/01/2015	Segunda Prueba de Evaluación Continua

Tabla 1: Calendario del plan de trabajo

3. Tecnologías utilizadas

A continuación se describen las tecnologías usadas en el desarrollo del proyecto, se han dividido en 2 áreas: tecnologías de hardware y tecnologías de software, se describen a continuación:

Tecnologías de Hardware

Cámara AXIS 211M. Es la fuente de imagen digital vía IP, esta cámara de red cuenta con un sensor de 1.3 mega-píxeles de alto rendimiento, esta diseñada para la vigilancia profesional por vídeo, proporciona imágenes claras y nítidas lo cual permite la identificación de objetos y personas [5].

Entre las características [5] más importantes de esta cámara están:

- Seguridad de red: Acceso multiusuario con protección por contraseña, filtrado de direcciones IP, cifrado HTTPS, Estándar IEEE 802.1X .
- Soporte IPv4 e IPv6.
- Manejo simultaneo de imágenes y vídeo en formatos JPEG y MPEG-4 respectivamente.
- Soporte de Quality of Service (QoS) .
- Interfaz de programación de aplicaciones (API) no usada en el proyecto.

Ilustración 1: Cámara AXIS 211M

Ilustración 2: Puertos de la Cámara AXIS 211M

Esta cámara fue adquirida teniendo en cuenta que es una de las cámaras más usadas en OpenDomo OS y con controladores ya probados en el mismo sistema operativo.

Computador. Se cuenta con un computador Toshiba Satellite 205, con sistema operativo Ubuntu 14.04, con un dispositivo de red y una conexión a Internet para descargar y configurar los diferentes paquetes necesarios en el proyecto, la cámara IP AXIS 211M estará conectada a un dispositivo de red de tipo Switch usando una dirección IP en el mismo rango que el computador.

Tecnologías de Software

OpenDomoOS 2.1. Es el sistema operativo de OpenDomo Services S.L esta diseñado para automatismos, domótica y eficiencia energética, está basado en un sistema de software libre desarrollado de forma participativa por la comunidad OpenDomo, OpenDomo cuenta entre otras con las siguientes características [1]:

- ***Soporte de Diferentes Protocolos***: permite usar y unificar diferentes tecnologías existentes en el área de la domótica, como uPnP, X10, EIB, etc, con el protocolo TCP/IP.

- ***Accesibilidad***: permite el control del hogar de forma sencilla, facilitando al usuario el uso de OpenDomo de forma intuitiva, integrando en una misma interfaz de PC, PDA o teléfono móvil los principales componentes del hogar, incluyendo sensores, actuadores, sistemas multimedia y sistemas de seguridad, entre otros.

- ***Seguridad*** : Es un sistema domótico seguro, estable y tolerante a fallos, por su naturaleza la seguridad física del usuario es prioritaria, ya que permite la apertura de puertas, persianas, sistemas de videovigilancia, alarmas, etc.

- ***Red de Agentes Distribuidos***: OpenDomo trabaja mediante una red de agentes distribuidos que es un sistema hardware que corre la distribución OpenDomo OS, cada agente se encarga de gestionar un conjunto de servicios de la red OpenDomo, siendo el responsable de los mismos y garantizando la tolerancia a fallos y continuidad de servicio sobre los dispositivos como cámaras IP, agentes OpenDomo, Sistemas Multimedia, placas de control, etc.

- ***Módulos OpenDomo***: El proyecto se organiza en varios módulos que juntos forman OpenDomo. El módulo principal es opendomo-distro. Este consiste en una distribución GNU/Linux cuyo objetivo principal es tener un reducido tamaño. Sobre opendomo-distro corren los demás módulos que ofrecen diferentes herramientas y acciones, extendiendo así las funcionalidades de la red domótica.

- ***Interfaz Gráfica***: La interfaz gráfica principal de OpenDomo se basa en

un sistema CGI el cual corre como un servicio en el agente opendomo-cgi que implementa a su vez un sistema de procesado de scripts permitiendo de forma sencilla crear aplicaciones. Estas aplicaciones son scripts que soportan cualquier tipo de lenguaje que pueda funcionar sobre GNU/Linux aunque el preferido es shellscript.

- ***Tecnología Base***: Los módulos que forman OpenDomo se desarrollan principalmente en C y shellscript. Opendomo-distro usa un kernel GNU/Linux y el conjunto de binarios BusyBox, la librería de C uclibc y su sistema buildroot.

Los recursos disponibles en la plataforma OpenDomo OS 2.1 y con los cuales se va a trabajar el proyecto son los siguientes:

- 1 Procesador en arquitecturas i686 y ARM
- 512 de RAM máximo 2GB
- 1 tarjeta de red
- Espacio en disco Tarjeta SD de 2GB

Además se usan los siguientes plug-ins de OpenDomo OS desde el repositorio de desarrollo, lo cual permite cambiar y ajustar los mismos al proyecto sin afectar la estabilidad del sistema operativo, estos plug-ins son [3]:

- ***opendomo-devel***: Este plug-in se instala desde el repositorio de desarrollo OpenDomo OS y proporciona herramientas de desarrollo útiles que permiten tener información visible desde el propio CGI de depuración y herramientas de traducción para adaptar el proyecto para cualquier idioma.

- ***opendomo-automation***: Este plug-in tiene scripts de automatización para OpenDomo OS 2.0, e implementa las características existentes de OpenDomo OS 1.0.0 en la versión 2.0, y trabaja en la adaptación de los scripts de comandos para las características del nuevo sistema base.

- ***opendomo-vision***: Este plug-in proporciona el soporte de procesamiento visual para cámaras IP en OpenDomo OS 2.0, permite reconocer la cámara en el sistema, registrarla en el sistema, tomar imágenes y guardarlas en un ciclo permanente.

VirtualBox. Es el entorno de virtualización con licencia GPL versión 2, se usara la versión suministrada por el repositorio de Ubuntu 14.04, este entorno de virtualización es necesario para poder usar OpenDomoOS 2.1 durante el proceso de desarrollo, permite cargar la máquina virtual suministrada por OpenDomo Services S.L desde el sitio web de OpenDomo [2].

OpenCV (Open Source Computer Vision Library). Es una biblioteca bajo una licencia BSD de código abierto creada por Intel, que incluye cientos de algoritmos de visión por computador para diferentes plataformas incluyendo GNU/Linux y usa diferentes lenguajes de programación, OpenCV tiene buena eficiencia computacional ya que su enfoque es hacia aplicaciones en tiempo real, lo que lo hace ideal para OpenDomo OS dados los escasos recursos que tienen las plataformas de control de este sistema operativo [6].

Estos algoritmos están distribuidos en diferentes paquetes de bibliotecas estáticas y dinámicas que forman los siguientes módulos [7]:

- **core o núcleo:** es un módulo compacto que define las estructuras de datos básicos, entre ellos los arreglos multidimensionales Mat y las funciones básicas utilizadas por el resto de módulos.

- **imgproc:** es un módulo de procesamiento de imagen que incluye filtrado de imagen lineal y no lineal, transformaciones geométricas de imagen, la conversión de espacio de colores, histogramas entre otros.

- **video:** es un módulo de análisis de vídeo que incluye estimación de movimiento, sustracción de fondo y algoritmos de seguimiento de objetos.

- **calib3d:** es un módulo que contiene algoritmos básicos para múltiples vistas, calibración de cámara, estimación de la postura de un objeto, algoritmos de correspondencia y elementos de reconstrucción 3D.

- **features2d:** es un módulo que permite el uso de detectores de características, descriptores y comparadores descriptores.

- **objdetect:** es un módulo de detección de objetos y clases predefinidas (por ejemplo: caras, ojos, gente, y carros entre otros).

- **highgui:** es una interfaz gráfica que permite usar fácilmente la captura de vídeo, imágenes y codecs de vídeo

- **gpu:** es un módulo que contiene algoritmos acelerados por GPU de diferentes módulos de OpenCV

También tiene otros módulos de ayuda como Flann, módulos de prueba y soporte a Python por medio de OpenCV-Python.

Python. Es un lenguaje de programación de alto nivel para propósito general, su filosofía de diseño hace énfasis en la legibilidad del código, su sintaxis permite a los programadores crear aplicaciones con un número menor de

líneas de código de lo que sería posible en los lenguajes de programación como C ++ o Java.

Python maneja varios paradigmas de programación, incluyendo programación orientada a objetos, programación imperativa y funcional o programación procedimental. Los intérpretes de Python están disponibles para diferentes plataformas, lo que permite la ejecución de código Python en la mayoría de los sistemas operativos.

Python es un software libre y de código abierto y tiene un modelo de desarrollo basado en su comunidad y es administrado por la organización no lucrativa de Python Software Foundation [8].

OpenCV-Python. Es una biblioteca de software libre y de código abierto que permite usar OpenCV en Python, OpenCV-Python hace uso de NumPy, que es una biblioteca optimizada para operaciones numéricas con una sintaxis de estilo MATLAB. Todas las estructuras de tipo matriz de OpenCV se convierten a matrices en NumPy, esto también hace que sea más fácil el integrarse con otras bibliotecas como SciPy and Matplotlib muy usadas en el tratamiento de imágenes [9].

Esta biblioteca es de vital importancia para el proyecto, ya que permitirá ejecutar scripts de Python usando OpenCv sobre OpenDomo OS, sistema operativo que trabaja desde su versión 2.1 con scripts interpretados haciendo de esta biblioteca la herramienta ideal para la programación de las funcionalidades OpenCV que se requieran.

Doxygen. es un generador de documentación publicado bajo los términos de la GNU Licencia Pública General, soporta múltiples lenguajes de programación, como C ++, C, C#, Objective-C, Java, Perl, Python, IDL, VHDL, Fortran y PHP.

GitHub. Para el control de versiones y seguimiento de los proyectos de OpenDomo Services S.L se usa GitHub, que usa Git para el control de versiones, pero GitHub también incluye navegador de código fuente, edición del código, wiki, seguimiento de tareas y fallos, permite definir el estado de cada tarea e hitos dentro del desarrollo del proyecto, en si es una forja, es gratis para código abierto y publico, ya que para proyectos privados hay que pagar por su uso.

Teniendo en cuenta lo anterior el proyecto para el desarrollo del plug-in usará GitHub, el sitio reservado para tal fin es: https://github.com/alherca/OpenCVODOS

GNU GENERAL PUBLIC LICENSE Versión 3. Para el plug-in se usa esta licencia, que es la misma que usa el sistema operativo OpenDomo OS 2.1, puede consultarse el contenido de la licencia en el Anexo 1.

4. Implementación

Adecuación de librerías en OpenDomoOS

En el apartado anterior se han visto las herramientas disponibles para el desarrollo del plug-in que en adelante se llamara OpenCVODOS sigla de OpenCV para OpenDomo OS, esas herramientas se ajustan a las necesidades y requerimientos de OpenDomo Services S.L, entre ellas y dentro del análisis previo realizado, el primer paso es poder instalar y usar correctamente las librerías necesarias de OpenCV en OpenDomo OS, para tal fin se debe contar con un entorno de trabajo que servirá también para el desarrollo del código fuente del plug-in.

Esta adecuación del entorno de trabajo requiere una serie de instalaciones y configuraciones que se describen a continuación:

1. Instalación de VirtualBox, se usa la versión desde el repositorio de Ubuntu 14.04 que es el sistema base del computador destinado para el desarrollo del proyecto, se usa el administrador de aplicaciones Synaptic, la versión soportada e instalada es VirtualBox 4.3.10_ubuntu.

2. Se realiza la descarga de la versión 2.1 de OpenDomo OS desde el sitio oficial de descargas http://es.opendomo.org/downloads, se obtiene el archivo en formato .ova (OpenDomoOS2.1.ova), que permite la importación directa y completa de la máquina virtual OpenDomoOS2.1 a VirtualBox [2].

3. Importación de la máquina virtual, para este paso usa la documentación oficial de OpenDomo al respecto [2] y describe la siguiente secuencia:

   - Se va al menú Archivo de VirtualBox
   - Se usa la opción Importar Servicio Virtualizado
   - Se abre el archivo descargado anteriormente llamado OpenDomoOS2.1.ova
   - Se da clic al botón Aceptar
   - Se procede a configurar la interfaz de red como adaptador puente
   - Se procede a desplegar el sistema OpenDomoOS.

   En este punto ya se cuenta con el sistema operativo completo que lo provee la máquina virtual de OpenDomo y se procede a entrar al mismo, hay que tener en cuenta que en el proceso de arranque VirtualBox le ha asignado una dirección IP por DHCP a la nueva máquina virtual, esa IP se podrá ver en pantalla cuando se esta cargando el sistema operativo [2].

Desde un navegador Web como se muestra en la ilustración 3, podemos entrar a la dirección 192.168.0.X donde X es el numero asignado al inicio de la máquina virtual, para tener acceso al sistema se usa el usuario: admin y como contraseña: opendomo y se obtendrá la siguiente pantalla:

Ilustración 3: Pantalla Web de OpenDomo OS

O también existe la posibilidad de entrar al sistema directamente por la consola usando como usuario: admin y como contraseña: opendomo, que son las credenciales por defecto usadas en OpenDomo OS, las cuales son cambiadas en el primer inicio del sistema.

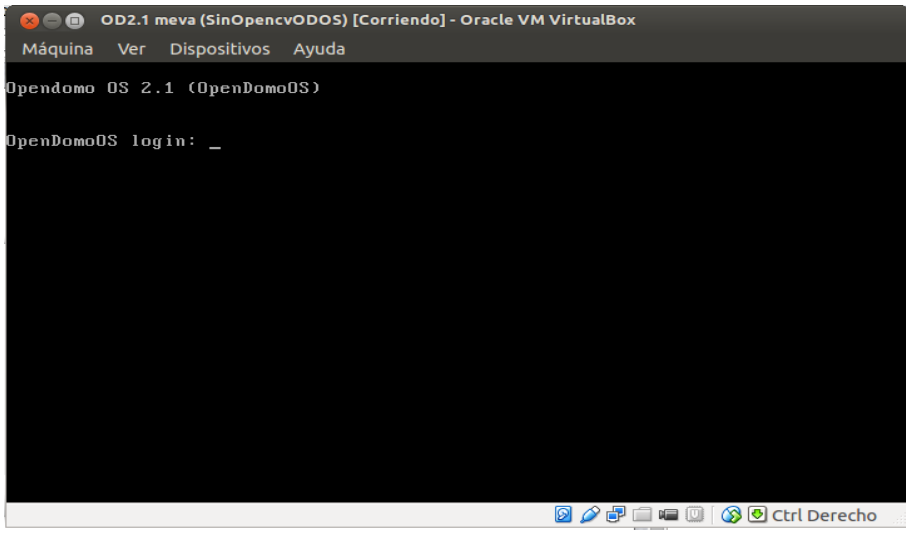

Ilustración 4: Consola de entrada de OpenDomo OS

4. Ahora se procede a instalar los plug-ins básicos de OpenDomo OS, el primero es el plug-in opendomo-devel, se hace desde el repositorio de desarrollo, para esto se usa el siguiente comando:

   ```
   plugin_add_from_gh.sh opalenzuela opendomo-devel
   ```

5. Instalación del plug-in opendomo-automation desde el repositorio de desarrollo, para esto se usa el siguiente comando:

   ```
   plugin_add_from_gh.sh opalenzuela opendomo-automation
   ```

6. Instalación del plug-in opendomo-vision desde el repositorio de desarrollo, para esto se usa el siguiente comando:

   ```
   plugin_add_from_gh.sh jmirasb opendomo-vision
   ```

7. Se adiciona la cámara AXIS IP en el sistema, para esto se ejecuta el siguiente comando:

   ```
   addControlDevice.sh http://192.168.0.90/ root sophia axis 5
   ```

   addControlDevice.sh permite adicionar dispositivos en el sistema en este caso requiere los siguientes parámetros:

   - Dirección IP de la cámara: http://192.168.0.90/
   - Usuario de la cámara: root (únicamente para pruebas)
   - Contraseña del usuario: sophia (únicamente para pruebas)
   - Tipo de dispositivo: axis
   - Tiempo de refresco: 5

8. Luego de esto se procede a reiniciar los plug-ins con los comandos:

   ```
   /usr/local/opendomo/daemons/odauto.sh restart
   /usr/local/opendomo/daemons/odvision.sh restart
   ```

9. Se verifica que se estén generando las imágenes (snapshot) de forma automática, estas imágenes quedan en la ruta /var/www/data con el nombre 192168090.jpg, el nombre de la imagen corresponde al ID que tiene el dispositivo en el sistema y que fue asignado al agregar el mismo al sistema, estas imágenes serán un insumo muy importante para la adecuación de los filtros usados por el plug-in que usará OpenCV.

10. Se realizaron pruebas de funcionamiento de OpenCV en un sistemas operativo Debian 7.0 sin interfaz gráfica, que es la distribución base de OpenDomo OS, con estas pruebas se pudo establecer los paquetes y dependencias necesarias para el correcto funcionamiento de OpenCV

con Python y se determino que se requieren 2 meta-paquetes, en el proceso de instalación estos son:

- libopencv-dev [11] que son los archivos de desarrollo de OpenCV y es requisito para el funcionamiento de OpenCV en el sistema.
- python-opencv [12] permite el uso de OpenCV desde Python

11. Para la instalación de estos meta-paquetes en el sistema OpenDomo OS 2.1, se configuro en GitHub el repositorio del proyecto, este contiene los archivos de instalación y configuración de plug-in, el repositorio como se dijo antes tiene el nombre de OpenCVODOS (https://github.com/alherca/OpenCVODOS).

    En el repositorio se adecuaron los siguientes archivos y directorios:

    - /usr/local/opendomo/filters contiene los directorios con los diferentes filtros en Python que serán usados por medio del plug-in
    - /usr/local/opendomo/bin/ contiene los archivos de configuración del plug-in
    - /var/opendomo/plugins/ contiene la definición y descripción de los paquetes necesarios que serán instalados por OpenDomo OS para que el plug-in OpenCVODOS pueda funcionar, específicamente en el archivo opencvodos.deps se adiciona la línea:

      libopencv-dev python-opencv

    - Con esto OpenDomo OS ya inicia la resolución de dependencias e instala los paquetes respectivos.

En la ilustración 5 se muestra la pantalla principal del repositorio publico de OpenCVODOS creado en GitHub, el cual contiene todo el código fuente del proyecto.

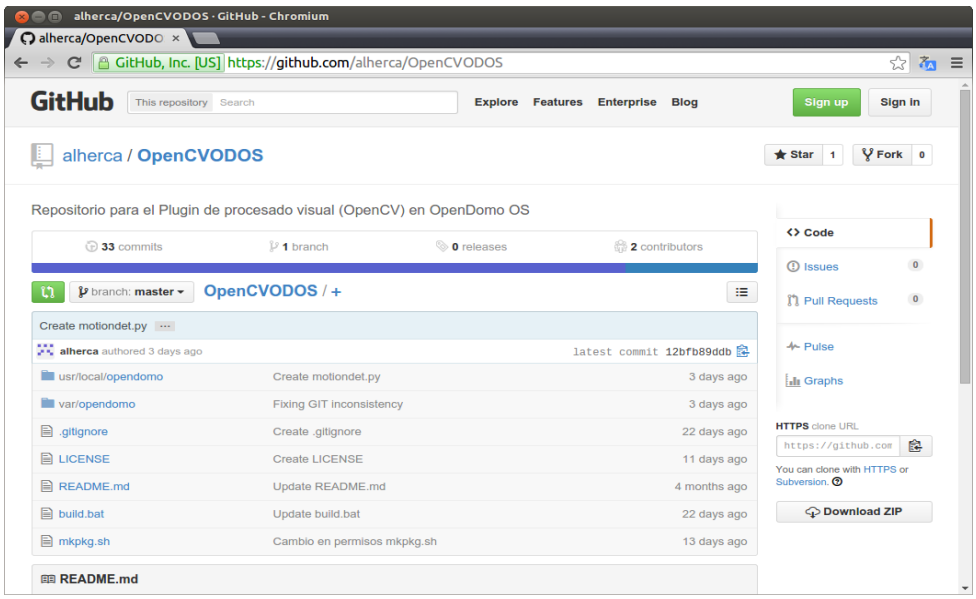

Ilustración 5: Repositorio OpenCVODOS

12. Con el repositorio configurado ya se procede a la instalación de OpenCVODOS en el sistema operativo OpenDomo OS 2.1, para esto usamos el siguiente comando:

    ```
    plugin_add_from_gh.sh alherca OpenCVODOS
    ```

    Debido a que los paquetes libopencv-dev y python-opencv requieren más de 308 MB se requiere reiniciar el sistema varias veces cuando este se queda sin espacio, para esto se usan 2 comandos de OpenDomo OS:

    - managePlugins.sh que muestra el estado de los plug-ins instalados y de ser necesario genera alarmas sobre el espacio, si muestra "#WARN no free space available" significa que no se han podido instalar más paquetes por falta de espacio, y se procede a usar el comando saveConfigReboot.sh.

    - saveConfigReboot.sh permite guardar la configuración actual y reiniciar el sistema, la ventaja en este procedimiento es que libera espacio que no es usado sin afectar el correcto funcionamiento de OpenDomo OS, esto permite continuar con la instalación de OpenCVODOS.

      Este procedimiento de reinicio del sistema será temporal ya que el equipo de OpenDomo Services S.L trabaja para que no sea necesario a futuro.

13. Luego se verifica que la variable de entorno PYTHONPATH este bien configurada con la ruta de las librerías necesarias por OpenCV, si no esta configurada será necesario ejecutar el siguiente comando:

```
export PYTHONPATH=/var/lib/python-support/python2.7/:$PYTHONPATH
```

14. Se procede a guardar la configuración con saveConfigReboot.sh, con este ultimo procedimiento ya se tiene adecuado el entorno y las librerías de OpenCV para el desarrollo del plug-in OpenCVODOS.

Diseño y Desarrollo del Plug-in OpenCVODOS

El diseño del plug-in OpenCVODOS esta definido por los requerimientos puntuales realizados por el equipo de OpenDomo Services S.L, estos son:

- opendomo-vision se hace cargo de configurar y visualizar cámaras, y sobre cada una de éstas cámaras trabajará el plug-in OpenCVODOS.

- OpenCVODOS será capaz de aplicar cada uno de los "filtros", que se desarrollen y están por definir en conjunto con el tutor de prácticas Oriol Palenzuela i Rosés.

- Los filtros tomarán como punto de partida una snapshot o imagen de la cámara, aplicará un algoritmo y generará una salida como imagen y/o una salida lógica que podrá lanzar eventos y/o ejecutar scripts, por ejemplo "movimiento detectado", "barrera virtual activada", etc.

- Para esto, se guardarán los archivos de configuración en el directorio /etc/opendomo/vision/ID_Camara/filters/, con un formato para cada archivo de configuración, y los scripts de gestión de los filtros.

- Para invocar los filtros desde el daemon de OpenCVODOS se definirán los datos necesarios como pueden ser ID del dispositivo, nombres de las imágenes y frecuencia de generación de los resultados, estos datos se almacenaran en un archivo en formato tipo texto.

Antes de describir la estructura y el diseño del plug-in cabe anotar que se realizaron pruebas previas ejecutando diferentes scripts en Python sobre Debian 7.0, estos scripts son los filtros que se adecuarán a OpenDomo OS y con los cuales OpenCVODOS trabajará.

Diseño del Plug-in OpenCVODOS

Teniendo en cuenta los objetivos del proyecto se ha diseñado la siguiente estructura para dar solución a los requerimientos del plug-in, estos requerimientos han sido previamente concertados con el equipo de OpenDomo Services S.L, el plug-in funciona de la siguiente forma y con los siguientes

componentes:

- opendomo-vision se encarga de configurar las cámaras y visualizar las mismas, opendomo-vision actualmente crea un ID por dispositivo, este ID se toma para el nombre de las imágenes generadas y guardadas en /var/www/data/.

- Por su parte OpencvODOS deberá aplicar por cada cámara el o los filtros seleccionados y activados, esto significa que deberá existir un mecanismo para guardar la configuración que contenga tenga el ID de la cámara donde se va a aplicar los filtros y los filtros que se aplicará a cada cámara , existe un archivo de configuración por cada cámara.

- Cada filtro estará desarrollado en Python para poder usar OpenCV y seria invocado desde un servicio de OpencvODOS, las imágenes necesarias serian suministradas por opendomo-vision, pero en algunos casos deberían crearse 2 imágenes en cada generación de imágenes, esto para hacer detección de movimiento o cambios entre imágenes.

- En algunos casos se pueden generar imágenes pero en otros se generaran salidas lógicas que podrán lanzar eventos y/o ejecutar scripts, por ejemplo "movimiento detectado", "barrera virtual activada", “persona detectada” etc, OpencvODOS corre como un servicio de OpenDomo OS y este ejecuta los filtros permanentemente, estos generan los eventos desde Python.

Estructura del Plug-in OpenCVODOS

Según la explicación anterior el plug-in OpenCVODOS tiene la siguiente estructura:

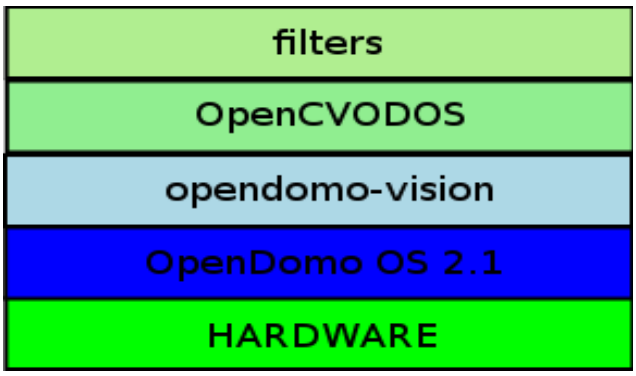

Ilustración 6: Estructura de OpenCVODOS

La estructura de archivos diseñada en el repositorio esta acorde a la arquitectura de OpenDomo OS y es la siguiente:

Directorio	Descripción	Archivos contenidos
/	Directorio raíz del plug-in	build.bat LICENSE mkpkg.sh README.md
/usr/local/opendomo/daemons/	Contiene el daemon de OpenCVODOS, desde donde se ejecutan todos los filtros configurados para las cámaras definidas.	opencvodos.sh
/usr/local/opendomo/events/	Contiene los literales correspondientes a cada evento.	opencvodos-motiondet opencvodos-fence
/usr/local/opendomo/filters/	Contiene los filtros en Python y están separados por directorios.	
/usr/local/opendomo/filters/circle/	Filtro de ejemplo que permite detectar círculos en las imágenes capturadas por las cámara configuradas.	circle.py
/usr/local/opendomo/filters/motiondet/	Filtro que permite detectar si hay un movimiento, usa 2 imágenes capturadas por las cámaras activadas.	motiondet.py
/usr/local/opendomo/filters/persons/	Filtro que permite detectar si hay una o varias personas, usa 1 imagen capturada por cada una de las cámaras activadas.	persons.py
/usr/local/opendomo/bin/	Contiene los scripts que permiten la configuración de las cámaras en OpenCVODOS y los filtros para las mismas	configureOpenCVODOS.sh addFilterToCamera.sh removeFilterToCamera.sh
/var/opendomo/plugins/	Contiene los archivos que permiten la instalación de las librerías y soporte Python de OpenCV desde el repositorio de OpenDomo OS.	opencvodos.deps opencvodos.desc opencvodos.info

Tabla 2: Estructura de Archivos del plug-in OpenCVODOS

Diagrama de Componentes del Plug-in OpenCVODOS

Los componentes relacionados según lo descrito anteriormente se pueden representar en el siguiente diagrama:

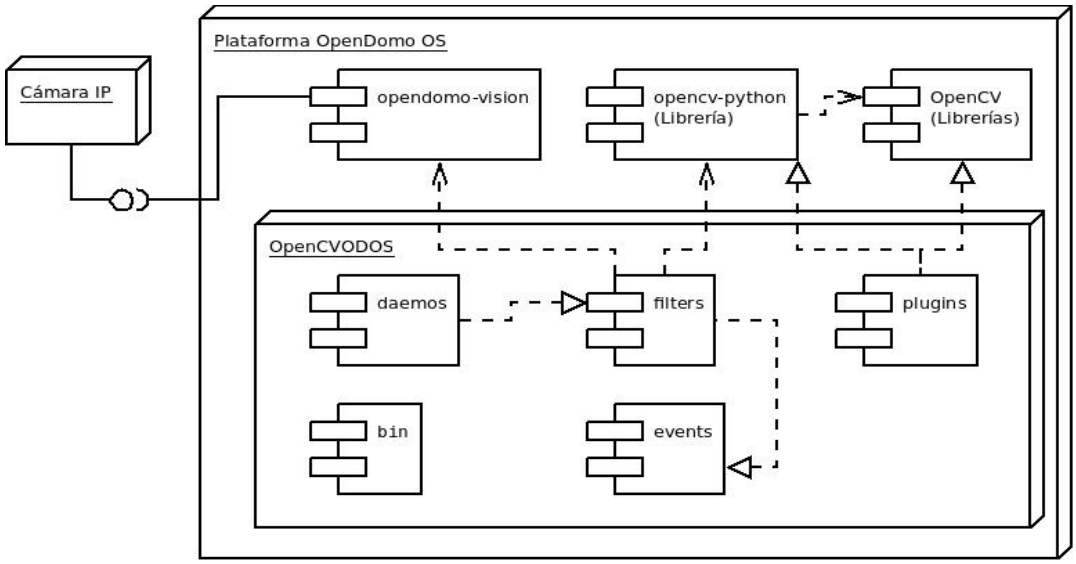

Ilustración 7: Diagrama de Componentes del Plug-in OpenCVODOS

Desarrollo del del Plug-in OpenCVODOS

Teniendo en cuenta el diseño descrito anteriormente se han definido los siguientes elementos con su respectiva explicación para dar funcionamiento al plug-in, estos elementos son:

Archivos Base

Son los archivos básicos que necesita OpenDomo 2.1 para instalar un plug-in y son plantillas suministradas por OpenDomo Services S.L, estos archivos son: build.bat, LICENSE, mkpkg.sh y README.md, estos archivos están ubicados en la raíz del repositorio del plug-in.

Archivos de Dependencias y librerías

Estos archivos ubicados en /var/opendomo/plugins/ son usados por OpenDomo 2.1 para descargar, configurar e instalar las dependencias y paquetes necesarios en un plug-in, son 3 archivos opencvodos.deps , opencvodos.desc , opencvodos.info, los 2 últimos archivos contienen la información y descripción de los paquetes, pero el archivo más importante es opencvodos.deps que contiene el nombre de las librerías y dependencias a instalar en el momento de invocar la instalación del plug-in OpenCVODOS, su contenido es el siguiente:

```
libopencv-dev python-opencv
```

Estos 2 paquetes permiten la instalación de OpenCV, Python y el soporte de OpenCV para Python.

configureOpenCVODOS.sh

Este script ubicado en /usr/local/opendomo/bin/ permite asignar una cámara previamente configurada en opendomo-vision a OpenCVODOS y realiza las siguientes tareas:

- Verifica que se entregue 2 argumentos, estos argumentos son:
    - ID de la cámara: que es el ID que ha generado opendomo-vision cuando se configuro la cámara en OpenDomo OS
    - Nombre a asignar: es el nombre con el cual se quiere identificar la asignación de la cámara a OpenCVODOS
- Si los argumentos no existen se entrega por pantalla el listado de cámaras configuradas en OpenDomo OS, además se provee una ayuda indicando como usar el comando.

- Si los argumentos son correctos se verifica la existencia de la cámara a configurar y se procede a crear un directorio con el nombre del ID de la cámara en /etc/opendomo/vision/ID_Camara/filters/, un ejemplo sería:
    - /etc/opendomo/vision/192168090/filters/
- Se crea en el directorio /etc/opendomo/vision/ un archivo de configuración de tipo texto con el nombre del ID de la cámara y de extensión .conf, un ejemplo sería:
    - /etc/opendomo/vision/192168090.conf
- El archivo 192168090.conf tendrá 2 líneas 1 con el ID de la cámara y otra linea con el nombre asignado permite guardar la asociación de una cámara a OpenCVODOS
- Si al verificar la existencia de la cámara esta no existe, se entrega en pantalla un mensaje indicando que no existe tal dispositivo.

La sintaxis para el uso del comando es la siguiente:

```
configureOpenCVODOS.sh ID_Camara NombreCamara
```

Para ejecutar el script una vez instalado el plug-in OpenCVODOS solo es necesario escribir en una terminal el siguiente comando:

```
configureOpenCVODOS.sh 192168090 camara1
```

El código generado para este script se puede ver en el repositorio del proyecto: https://github.com/alherca/OpenCVODOS/blob/master/usr/local/opendomo/bin/configureOpenCVODOS.sh

addFilterToCamera.sh

Este script ubicado en /usr/local/opendomo/bin/ permite asignar o activar uno o varios filtros a OpenCVODOS, previamente debe ser asignado OpenCVODOS a una o varias cámaras con el script configureOpenCVODOS.sh, addFilterToCamera.sh realiza las siguientes tareas:

- Verifica que se entreguen los argumentos necesarios para cada filtro, estos argumentos son:
    - ID de la cámara: que es el ID que ha configurado en el plug-in OpenCVODOS

    - Nombre del filtro o de los filtros a asignar: es el nombre del filtro que se quiere asignar a la cámara.

- Si los argumentos no existen se entrega por pantalla el listado de cámaras configuradas en OpenDomo OS, además se provee una ayuda indicando como usar el comando.

- Si los argumentos son correctos se verifica la existencia de la cámara ya configurada para el plug-in OpenCVODOS y se procede a verificar la existencia del archivo de configuración generado por configureOpenCVODOS.sh con esto se garantiza que ya hay asignación de una determinada cámara al pulg-in el directorio /etc/opendomo/vision/.

- En el directorio /etc/opendomo/vision/ID_Camara/filters/ se crea un archivo de configuración de tipo texto, como contenido tendrá:

    - El encabezado "[Definition]" para uso posterior en Python

    - El nombre del ID de la cámara

    - El nombre del filtro a usar

- Este archivo tiene una extensión .conf, quedará de la siguiente forma /etc/opendomo/vision/ID_Camara/filters/NombreFiltro.conf, un ejemplo sería:

    - /etc/opendomo/vision/192168090/filters/motiondet.conf

- Si al verificar la existencia de la cámara esta no existe, se entrega en pantalla un mensaje indicando que no existe tal dispositivo asignado a OpenCVODOS.

La sintaxis para el uso del comando es la siguiente:

```
addFilterToCamera.sh ID_Camara NombreFiltro [NombresFiltros]
```

Para ejecutar el script una vez instalado el plug-in OpenCVODOS solo es necesario escribir en una terminal el siguiente comando:

```
addFilterToCamera.sh 192168090 motiondet
```

- Para asignar o activar varios filtros:

```
addFilterToCamera.sh 192168090 motiondet circle persons
```

El código generado para este script se puede ver en el repositorio del proyecto: https://github.com/alherca/OpenCVODOS/blob/master/usr/local/opendomo/bin/addFilterToCamera.sh

removeFilterToCamera.sh

Este script ubicado en /usr/local/opendomo/bin/ permite quitar o desactivar uno o varios filtros asignados previamente a OpenCVODOS, removeFilterToCamera.sh realiza las siguientes tareas:

- Verifica que se entreguen los argumentos necesarios para cada filtro, estos argumentos son:
    - ID de la cámara: que es el ID que ha configurado en el plug-in OpenCVODOS
    - Nombre del filtro o de los filtros a quitar: es el nombre del filtro que se quiere desactivar a la cámara.
- Si los argumentos no existen se entrega por pantalla el listado de cámaras configuradas en OpenDomo OS, además se provee una ayuda indicando como usar el comando.
- Si los argumentos son correctos se verifica la existencia del filtro con el archivo de configuración respectivo, y luego se procede a borrar el archivo que corresponda en el directorio: /etc/opendomo/vision/ID_Camara/filters/, con esta acción se quita la asignación del filtro a la cámara.
- Si al verificar la existencia de la cámara esta no existe, se entrega en pantalla un mensaje indicando que no existe tal dispositivo asignado a OpenCVODOS.

La sintaxis para el uso del comando es la siguiente:

```
removeFilterToCamera.sh ID_Camara NombreFiltro [NombresFiltros]
```

Para ejecutar el script una vez instalado el plug-in OpenCVODOS solo es necesario escribir en una terminal el siguiente comando:

```
removeFilterToCamera.sh 192168090 motiondet
```

- Para quitar o desactivar varios filtros:

```
removeFilterToCamera.sh 192168090 motiondet circle persons
```

El código generado para este script se puede ver en el repositorio del proyecto: https://github.com/alherca/OpenCVODOS/blob/master/usr/local/opendomo/bin/removeFilterToCamera.sh

opencvodos.sh

Este script ubicado en /usr/local/opendomo/daemons/ es el daemon de OpenCVODOS, permite ejecutar periódicamente todos los filtros de todas las cámaras configuradas para OpenCVODOS, realiza las siguientes tareas:

- Carga el entorno de trabajo para Python con la linea:

```
export PYTHONPATH=/var/lib/python-support/python2.7/:$PYTHONPATH
```

- Verifica y hace un listado de las cámaras asociadas a OpenCVODOS, esto por medio de los archivos .conf de /etc/opendomo/vision/.

- Luego hace un listado de los filtros asociados a cada cámara del listado anterior, y ejecuta cada uno de esos filtros periódicamente de acuerdo a un tiempo definido de refresco en el mismo script.

- Provee las opciones de start, stop, status, restart y force-reload, típicas en los daemos para OpenDomo OS.

Este script se ejecuta automáticamente como un daemon, pero puede ejecutarse manualmente, solo es necesario escribir en una terminal el siguiente comando:

```
/usr/local/opendomo/daemons/opencvodos.sh start
```

El código generado para este script se puede ver en el repositorio del proyecto: https://github.com/alherca/OpenCVODOS/blob/master/usr/local/opendomo/daemons/opencvodos.sh

Archivos de literales

Estos archivos ubicados en /usr/local/opendomo/events/ contienen los literales correspondientes a un determinado evento, estos archivos son:

- opencvodos-motiondet: contiene el literal “Motion detected in Camera”, para indicar que se detecto un movimiento en una determinada cámara.

- opencvodos-person: contiene el literal “Person detected in Camera”, para indicar que se detecto una persona en una determinada cámara.

- opencvodos-circle: contiene el literal “Detected circle in Camera”, para indicar que se detecto un circulo en una determinada cámara.

Filtro circle.py

Es un script en Python que permite detectar círculos en una imagen [10][13] [14], es un script para pruebas se dejará implementado en el repositorio con ese fin, realiza las siguientes tareas:

- Recibe un argumento que es el ID de la cámara donde se aplica este filtro, este argumento fue enviado desde el daemon de OpenCVODOS.

- Carga el archivo de configuración del filtro llamado circle.conf, que contiene un formato para ser leído desde ConfigParser, una librería de Python especializada en leer archivos de configuración, este archivo esta en /etc/opendomo/vision/ID_Camara/filters/circle.conf, este archivo tiene el ID de la cámara, el nombre y otros posibles parámetros que pueden variar según el filtro.

- Luego lee la imagen desde /var/www/data/IDCamara.jpg

- Este archivo es procesado por el filtro, en este caso buscando círculos en la imagen, los cuales se resaltan en un circulo de color verde y un cuadrado de color naranja en el centro del circulo.

- Si se encuentran círculos se procede a generar un evento de log con el comando de OpenDomo OS llamado logevent y se captura y almacena en el archivo de log de eventos en /var/opendomo/log/events.log.

- Luego se ejecuta un evento físico que consiste en guardar una imagen con los círculos detectados en /var/www/data/ con el nombre IDCamara_circle.png.

En la Ilustración 8 se muestra una imagen antes de ser procesada por el filtro circle.py, en la Ilustración 9 se muestra la imagen resultante después del procesado realizado por el filtro circle.py, esta ultima imagen es almacenada en /var/www/data/ con el nombre ID_Camara_circle.png, donde ID_Camara es el ID de la cámara en la cual se aplica el filtro.

Ilustración 8: Captura antes del procesado

Ilustración 9: Procesado por circle.py

El código generado para este script se puede ver en el repositorio del proyecto: https://github.com/alherca/OpenCVODOS/blob/master/usr/local/opendomo/filters/circle/circle.py

Filtro motiondet.py

Es un script en Python que permite detectar movimiento comparando 2 imágenes capturadas por cada una de las cámaras configuradas en OpenDomo OS [10][13][14], realiza las siguientes tareas:

- Recibe un argumento que es el ID de la cámara donde se aplica este filtro, este argumento fue enviado desde el daemon de OpenCVODOS.

- Carga el archivo de configuración del filtro llamado motiondet.conf, que también contiene un formato para ser leído desde ConfigParser, la librería de Python especializada en leer archivos de configuración, este archivo esta en /etc/opendomo/vision/ID_Camara/filters/motiondet.conf, este archivo tiene el ID de la cámara, el nombre y otros posibles parámetros que varían según el filtro.

- Luego lee 2 imágenes generadas con opendomo-vision desde /var/www/data/ una imagen se llama ID_Camara.jpg y la otra tendrá el nombre de prev_ID_Camara.jpg

- Estos 2 archivos son procesados por el filtro, en este caso buscando las diferencias entre las 2 imágenes.

- Si se encuentran diferencias significa que existe un movimiento y se procede a generar un evento de log con el comando de OpenDomo OS

llamado logevent, además se captura y almacena en el archivo de log de eventos en /var/opendomo/log/events.log.

- Luego se ejecuta un evento físico que consiste en guardar una imagen con la imagen del movimiento detectado en /var/www/data/ con el nombre IDCamara_motiondet.png.

Ilustración 10: Imagen prev_192168090.jpg

Ilustración 11: Imagen 192168090_motiondet.png

En la Ilustración 10 se muestra un ejemplo de la imagen previa a la detección de movimiento llamada prev_192168090.jpg, luego en la Ilustración 11 se muestra la imagen resultante de la detección de movimiento creada con motiondet.py, la imagen tiene como nombre IDCamara_motiondet.png, donde IDCamara es 192168090 como resultado el nombre de la imagen es 192168090_motiondet.png.

El código generado para este script se puede ver en el repositorio del proyecto: https://github.com/alherca/OpenCVODOS/blob/master/usr/local/opendomo/filters/motiondet/motiondet.py

Filtro persons.py

Es un script en Python que permite detectar una o varias personas en una imagen capturada por cada una de las cámaras configuradas en OpenDomo OS [10][13][14], realiza las siguientes tareas:

- Recibe un argumento que es el ID de la cámara donde se aplica este filtro, este argumento fue enviado desde el daemon de OpenCVODOS.

- Carga el archivo de configuración del filtro llamado persons.conf, que

sigue aplicando el mismo procedimiento para leerse desde ConfigParser, la librería de Python especializada en leer archivos de configuración, este archivo esta en /etc/opendomo/vision/ID_Camara/filters/persons.conf, este archivo tiene el ID de la cámara, el nombre y otros posibles parámetros que varían según el filtro.

- Luego lee 1 imagen generada con opendomo-vision desde /var/www/data/ esta imagen se llama ID_Camara.jpg.

- Esta imagen es procesada por el filtro haciendo un barrido de toda la imagen buscando a las personas que se encuentren en la misma, cuando las detecta las encierra en un rectángulo.

- Si se encuentran personas se procede a generar un evento de log con el comando de OpenDomo OS llamado logevent, además se captura y almacena en el archivo de log de eventos en /var/opendomo/log/events.log.

- Luego se ejecuta un evento físico que consiste en guardar una imagen con la imagen del movimiento detectado en /var/www/data/ con el nombre IDCamara_persons.png.

Ilustración 12: Imagen 192168090.jpg

Ilustración 13: Imagen 192168090_persons.png

En la Ilustración 12 se muestra un ejemplo de la imagen antes de ser procesada por el filtro persons.py, en la Ilustración 13 se muestra la imagen creada por persons.py, la imagen muestra las personas detectadas encerradas en un rectángulo por cada persona, el archivo tiene como nombre IDCamara_persons.png, donde IDCamara es 192168090 como resultado el nombre de la imagen es 192168090_persons.png.

El código generado para este script se puede ver en el repositorio del proyecto: https://github.com/alherca/OpenCVODOS/blob/master/usr/local/opendomo/filters/persons/persons.py

Repositorio del código

La totalidad del código fuente del plug-in OpenCVODOS esta en el repositorio del proyecto: https://github.com/alherca/OpenCVODOS

5. Pruebas y validación del plug-in OpenCVODOS

Para realizar las pruebas se ha establecido en primera instancia un procedimiento de instalación y ejecución del plug-in en OpenDomo OS, este se describe a continuación:

1. Requerimientos previos:

    - Tener acceso a un sistema OpenDomo OS

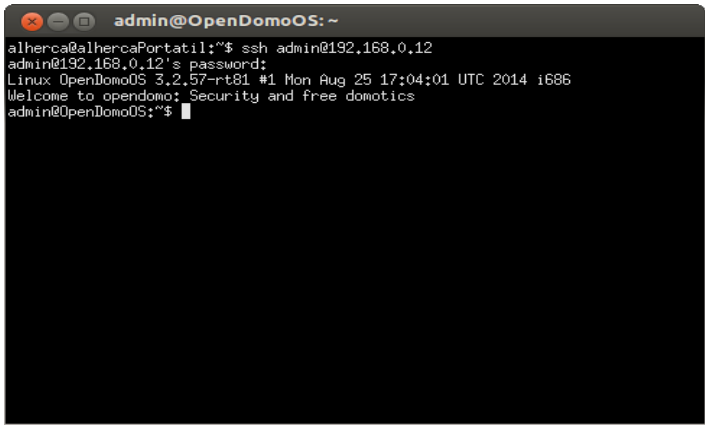

Ilustración 14: Acceso a un sistema OpenDomo OS

    - Tener instalado opendomo-vision, usar:

```
plugin_add_from_gh.sh jmirasb opendomo-vision
```

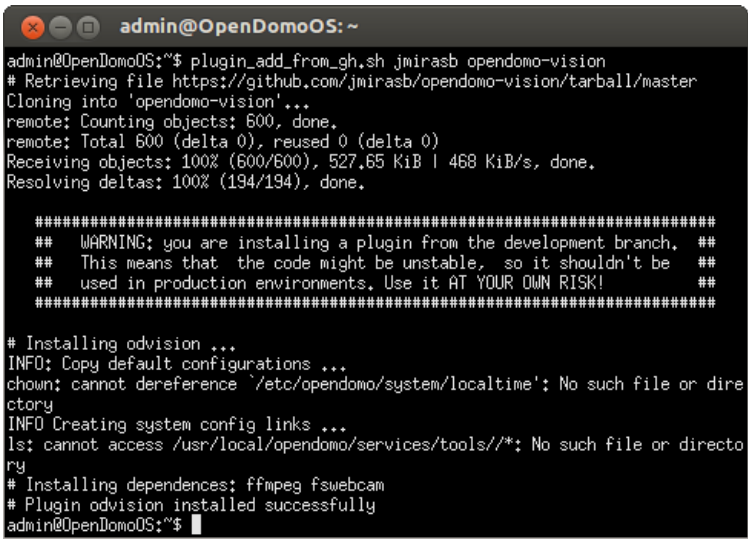

Ilustración 15: Instalación de opendomo-vision

- Tener instalado opendomo-automation, usar:

```
plugin_add_from_gh.sh opalenzuela opendomo-automation
```

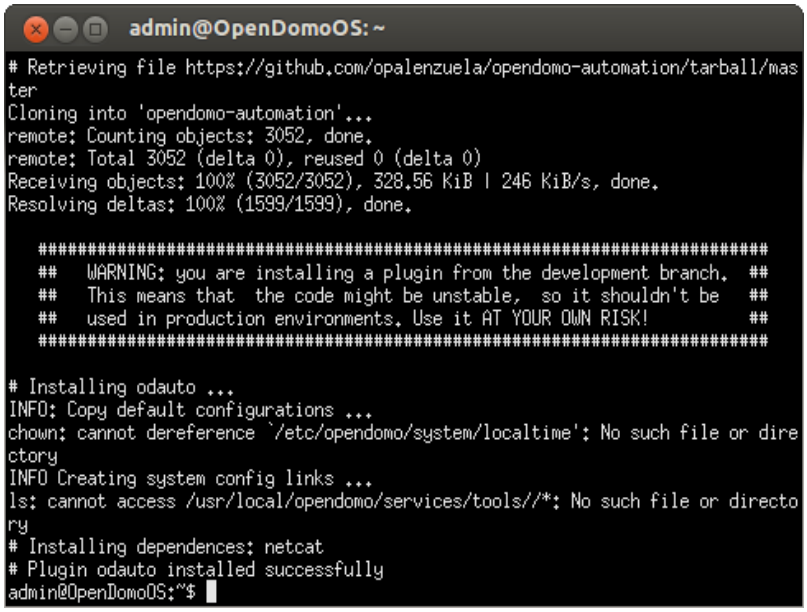

Ilustración 16: Instalación de opendomo-automation

- Instalar al menos una cámara IP en OpenDomo OS, se puede usar:

```
addControlDevice.sh IP User Password TipoCamara Refresco
```

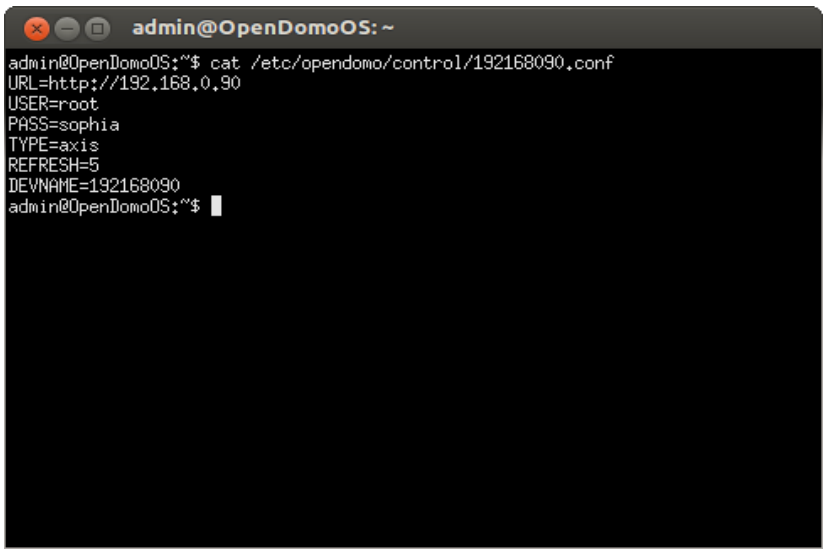

Ilustración 17: Cámara IP instalada en OpenDomo OS

- Desde la consola de OpenDomo OS se ejecutan los siguientes comandos.

2. Instalación de OpenCVODOS desde el repositorio:

   ```
   plugin_add_from_gh.sh alherca OpenCVODOS
   ```

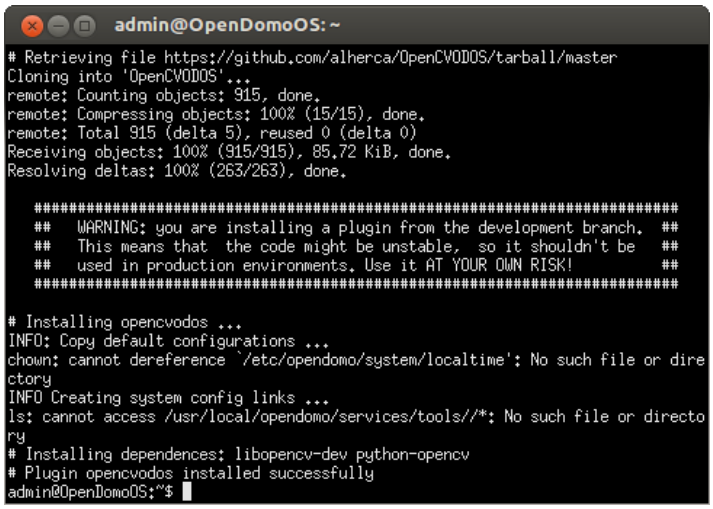

Ilustración 18: Instalación de OpenCVODOS

3. Asignar la cámara a OpenCVODOS con:

   ```
   configureOpenCVODOS.sh 192168090 camara1
   ```

   Se asume que 192168090 es el ID de la cámara

4. Se asignan los filtros a la cámara con:

   ```
   addFilterToCamera.sh 192168090 motiondet circle persons
   ```

5. Se verifica si existen o no los archivos de configuración con:

   ```
   ls /etc/opendomo/vision/192168090/filters/
   ```

6. Se hacen pruebas para quitar los filtros de la camara con:

   ```
   removeFilterToCamera.sh 192168090 motiondet circle persons
   ```

7. Se verifica si existen o no los archivos de configuración con:

   ```
   ls /etc/opendomo/vision/192168090/filters/
   ```

8. Se adiciona los filtros nuevamente a la cámara con:

   ```
   addFilterToCamera.sh 192168090 motiondet circle persons
   ```

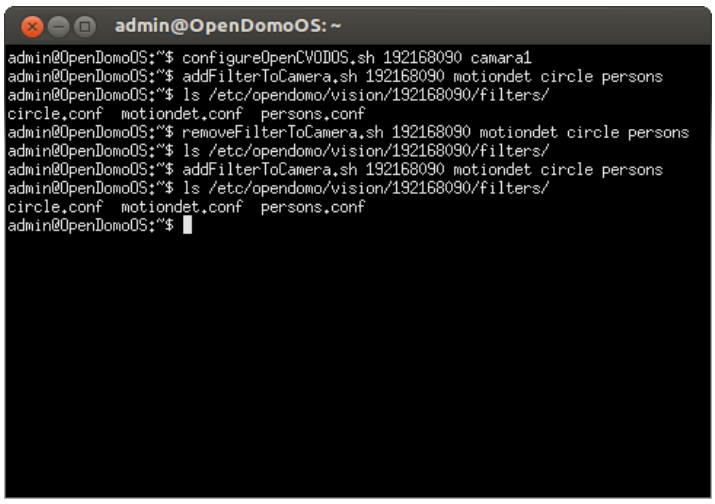

Ilustración 19: Pruebas de activación de OpenCVODOS y filtros en OpenDomoOS

9. Se arrancan o reinician los servicios relacionados con:

```
/usr/local/opendomo/daemons/odauto.sh restart
/usr/local/opendomo/daemons/odvision.sh restart
/usr/local/opendomo/daemons/opencvodos.sh restart
```

10. Ahora se verifica las salidas en /var/www/data/ con:

```
ls /var/www/data/
```

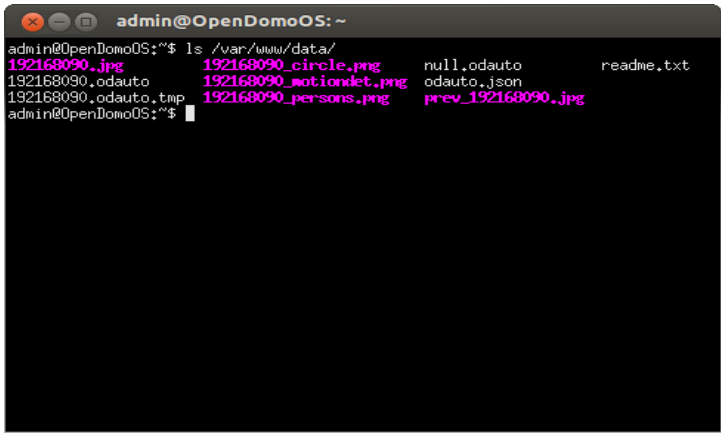

Ilustración 20: Verificación de imágenes de salida

11. Se verifica cada imagen y los resultados de cada una de ellas.

Batería de pruebas

Se ha propuesto una batería de pruebas para ser ejecutada por Opendomo Services S.L., y es descrita en la siguiente tabla:

Prueba	Test	Descripción	Resultado
1	1	Instalación del plug-in con: plugin_add_from_gh.sh alherca OpenCVODOS	Satisfactorio
2	1	Configurar OpenCVODOS parámetros correctos usando configureOpenCVODOS.sh	Satisfactorio
	2	Configurar OpenCVODOS parámetros incorrectos usando configureOpenCVODOS.sh	Satisfactorio
3	1	Asignar o activar filtro con addFilterToCamera.sh usando parámetros correctos	Satisfactorio
	2	Asignar o activar filtro con addFilterToCamera.sh usando parámetros incorrectos	Satisfactorio
4	1	Quitar o desactivar filtro con removeFilterToCamera.sh usando parámetros correctos	Satisfactorio
	2	Quitar o desactivar filtro con removeFilterToCamera.sh usando parámetros incorrectos	Satisfactorio
5	1	Ejecutar el daemon opencvodos.sh con parámetros correctos	Satisfactorio
	2	Ejecutar el daemon opencvodos.sh con parámetros incorrectos	Satisfactorio
6	1	Evento Físico filtro circle.py	Satisfactorio
	2	Evento Lógico filtro circle.py	Satisfactorio
7	1	Evento Físico filtro motiondet.py	Satisfactorio
	2	Evento Lógico filtro motiondet.py	Satisfactorio
8	1	Evento Físico filtro persons.py	Satisfactorio
	2	Evento Lógico filtro persons.py	Satisfactorio
9	1	Detener el daemon opencvodos.sh con parámetros correctos	Satisfactorio
	2	Detener el daemon opencvodos.sh con parámetros incorrectos	Satisfactorio

Tabla 3: Batería de pruebas

Los resultados mostrados son ejecutados por ahora por el desarrollador.

6. Conclusiones

Respecto a los objetivos propuestos a continuación se hace un análisis sobre el cumplimiento de los mismos:

- Se logró desarrollar el plug-in llamado OpenCVODOS, que permite usar el sistema de procesado de imagen OpenCV desde OpenDomo OS , las imágenes necesarias son suministradas por el plug-in de procesamiento visual opendomo-vision de OpenDomo OS, el nuevo plug-in OpenCVODOS usa las cámaras IP configuradas por opendomo-vision , instala y usa las librerías de OpenCV , además a esto se desarrollaron filtros para probar algunas funcionalidades de OpenCV.
- Se logro conocer el funcionamiento teórico y práctico de OpenCV en GNU/Linux, gracias a las pruebas realizadas sobre GNU/Linux Debian, esto permitió definir que paquetes se necesitan para poder ejecutar y como usar en OpenDomo OS las librerías de OpenCV desde Python.
- Se encontró como integrar las librerías de OpenCV y de Python en OpenDomo OS, de manera tal que el plug-in desarrollado instala los paquetes y las dependencias necesarias para el funcionamiento del mismo, esto desde el repositorio de código de OpenDomo OS.
- Se diseñó y desarrolló el plug-in construyendo el código fuente en el repositorio del proyecto, los diferentes comandos fueron integrados a OpenDomo OS, de manera tal que el usuario pueda relacionar las cámaras existentes con el plug-in y las herramientas que tengan los mismos, además el plug-in funciona como un servicio más de OpenDomo OS.
- Se documentó el código fuente con la herramienta Doxygen, además se documentaron las diferentes instrucciones de uso y las explicaciones relevantes referentes al código fuente desarrollado, lo que permitirá a futuro continuar con el desarrollo e incorporar nuevas herramientas en torno al plug-in y la posible publicación de esta versión beta en futuras versiones o actualizaciones de OpenDomo SO.
- Se elaboró una batería de pruebas que permitió garantizar el buen funcionamiento del plug-in en OpenDomo OS, validando diferentes situaciones típicas en la configuración y puesta en marcha del plug-in.

7. Seguimiento del cronograma de trabajo

Como se mostró en la planificación de la sección 2 de objetivos de esta memoria, el desarrollo del proyecto tiene una duración de un semestre, este tiempo y fechas se cumplieron, se describen a continuación los principales hitos con los problemas encontrados y la solución a los mismos.

Inicio Trabajo Final de Máster: no existieron contratiempos, la información por parte de la UOC, el consultor y el tutor de prácticas externo siempre fue oportuna y precisa desde el inicio, hay que destacar que desde el momento de la asignación del proyecto en mayo de 2014 se empezaron a recibir indicaciones por parte del consultor externo, estas indicaciones fueron muy oportunas ya que permitieron adelantar algunos trabajos para conocer OpenDomo OS y OpenCV.

Crear plan de trabajo: se planeo el trabajo a lo largo del semestre y se genero un documento que fue revisado por el consultor y el tutor de prácticas externas, este documento fue entregado en la PEC 1 sin contratiempos.

Estudiar el funcionamiento de OpenDomoOS: se realiza con la asesoría de OpenDomo Services S.L el estudio básico del funcionamiento de OpenDomo OS, se siguió la documentación oficial [2], y se empezaron a estudiar los mecanismos de funcionamiento del sistema operativo, se hizo énfasis en entender los siguientes aspectos:

- La gestión del espacio y recursos disponibles en la plataforma.
- La gestión de los plug-ins en OpenDomo OS, esto incluye instalación de un plug-in, instalación de paquetes, solución de dependencias.
- La gestión de los servicios o daemons en OpenDomo OS.
- La gestión de logs en la plataforma.

No se encontraron mayores problemas y las dudas fueron resueltas por el tutor de prácticas externas de la mejor manera.

Estudiar el funcionamiento de OpenCV: se consultaron las principales fuentes sobre OpenCV incluyendo la documentación oficial [6] y algunos libros que tratan sobre estas librerías [7][10], con este paso se pudo entender el funcionamiento general de OpenCV, también se estableció como estas librerías se pueden usar desde Python [9].

Integrar librerías existentes para ser usados desde el plug-in: para este paso por recomendación del tutor de prácticas externas se empezó realizando pruebas de instalación de paquetes en un sistema Debian 7.0 sin interfaz gráfica, se aclara que Debian es el sistema base de donde se deriva OpenDomo OS, en este contexto se determinaron los paquetes y

dependencias que necesita OpenCV y Python para funcionar correctamente, se ejecutaron algunos scripts de ejemplo [14] sin problemas, en este punto existió un problema que fue el poco espacio en disco de OpenDomo OS frente a todas las dependencias y espacio necesario por OpenCV y Python, se buscaron 2 alternativas de solución que se describen a continuación:

1. Compilación de OpenDomo OS "in situ" con las librerías incorporadas en el mismo, esta solución se probo, se obtuvo como resultado una imagen del gran tamaño y fue descartada dados los escasos recursos que tienen las plataformas de control de OpenDomo, y en especial Raspberry PI, además a partir de la versión 2 de OpenDomoOS, cualquier bloque de código no interpretado tiene que ser descargada mediante apt-get.

2. Solucionar el problema de dependencias en OpenDomo OS, esto corrió a cargo del grupo de trabajo de OpenDomo Service S.L, la manera suministrada como solución fue usar los comandos managePlugins.sh y saveConfigReboot.sh.
   El comando managePlugins.sh muestra el estado de los plug-ins instalados y de ser necesario genera alarmas sobre el espacio indicando que no se ha podido instalar más paquetes por falta de espacio, y el comando saveConfigReboot.sh permite guardar la configuración actual incluyendo paquetes instalados hasta el momento y reiniciar el sistema, liberando espacio que no es usado sin afectar el correcto funcionamiento de OpenDomo OS

De esta forma se instalaron los paquetes y dependencias para el correcto funcionamiento en OpenDomo OS de OpenCV integrado a Python.

Diseñar y desarrollar el plug-in: en este punto se encontraron problemas de ubicación de archivos, generación de imágenes y scripts, estos problemas retrasaron un poco el cronograma, pero por fortuna se pudo adelantar el trabajo respectivo para estar al día con la planificación definida desde el principio del proyecto, algunos de estos problemas se describen a continuación junto con su respectiva solución:

- Repositorio e instalación del plug-in, se realizaron varias correcciones al archivo mkpkg.sh que contiene los comandos de empaqueta-miento y destino del plug-in, garantizando que los destinos de los archivos del mismo sean los apropiados en la instalación sobre OpenDomo OS.

- Permisos en el repositorio, se encontró problemas de permisos de ejecución en algunos scripts ya que en el repositorio por defecto los archivos no tienen estos permisos, para esto se encontró la siguiente secuencia de comandos desde una terminal de GNU/Linux como solución:

```
git clone https://github.com/alherca/OpenCVODOS
cd OpenCVODOS/
git remote -v
git push origin master
git ls-tree HEAD
git update-index --chmod=+x archivo_a_cambiar_permisos.sh
git status
git config --global user.email "alherca.co@gmail.com"
git commit -m "Cambio en permisos mkpkg.sh"
git ls-tree HEAD
git push origin master
```

- Ubicación de scripts, al inicio se dejaron algunos de los scripts desarrollados en directorios que no correspondían a la ubicación óptima y ordenada en el OpenDomo OS, después de una revisión por parte del tutor de prácticas externas se reorganizaron los archivos creando los directorios respectivos y dándole al plug-in el orden requerido por el sistema operativo.

- Generación de imágenes, se encontraron algunos problemas en el desarrollo de los filtros sobretodo en la generación de la imagen resultante de cada uno de ellos, gracias a los libros consultados [10][14] se logro obtener el resultado esperado.

- Configuración de la cámara IP, se realizaron múltiples pruebas con una cámara IP FOSCAM y con una cámara USB, pero con el driver existente en ese momento no se pudo obtener las imágenes necesarias para el desarrollo del proyecto dentro de OpenDomo OS, para solucionar esto se adquirió la cámara AXIS 211M con la que si se pudo obtener las imágenes necesarias y continuar con el proyecto.

- Generación de imágenes desde opendomo-vision, se encontró un problema en la generación de las imágenes por parte de opendomo-vision, este problema se debía a que el archivo bind_axis.sh no tenia permisos de ejecución y la solución fue hacer la corrección respectiva en el repositorio de opendomo-vision desde GitHub, aun así el error continuaba, pero se logro detectar que además de los permisos existía una linea errónea en el script bind_axis.sh, en lugar de:

  ```
  $FULLURL="$URL/axis-cgi/jpg/image.cgi?resolution=320x240"
  ```

  debería ser:

  ```
  FULLURL="$URL/axis-cgi/jpg/image.cgi?resolution=320x240"
  ```

después de esto ya se obtuvieron las imágenes correctamente y se pudo continuar con el proyecto.

- En cuanto el desarrollo el ítem de más complejidad fue el leer archivos de configuración en Python, esto se soluciono estudiando la librería ConfigParser, una librería de Python especializada en leer archivos de configuración, permitiendo leer las variables del archivo como una variable más en Python.

- Otro problema encontrado fue la ejecución de comandos tipo shell desde Python, esto se soluciono con la librería subprocess, que permite enviar como parámetro el comando a ejecutar en el sistema operativo.

Pruebas y validación de la aplicación: no se encontraron problemas en este sentido y se pudieron ejecutar las pruebas satisfactoriamente.

Crear documentos necesarios para la entrega final: se pudo adelantar la documentación satisfactoriamente, pero hubiese sido mejor dejar mas tiempo para esta parte en la planificación, ya que el trabajo se intensifico considerablemente al final del proyecto para poder terminar la documentación en la fecha planeada.

Crear presentación del trabajo final: se creo la presentación sin contratiempos.

Entrega PEC 2 Trabajo final: se entrega el proyecto en la fecha planeada.

En el siguiente diagrama de Gantt de la tabla 4 se muestran las tareas y su duración.

WBS	Nombre	Trabajo
1	Inicio Trabajo Final de Máster	1d
2	Crear plan de trabajo	9d
3	Entrega PEC 1 Plan de trabajo	1d
4	Estudiar el funcionamiento de OpenDomoOS	18d
5	Estudiar el funcionamiento de OpenCV	21d
6	Integrar librerías existentes para ser usados desde el plugin	28d
7	Diseñar y desarrollar el plugin	25d
8	Pruebas y validación de la aplicación	6d
9	Crear documentos necesarios para la entrega final	10d
10	Crear presentación del trabajo final	10d
11	Entrega PEC 2 Trabajo final	1d

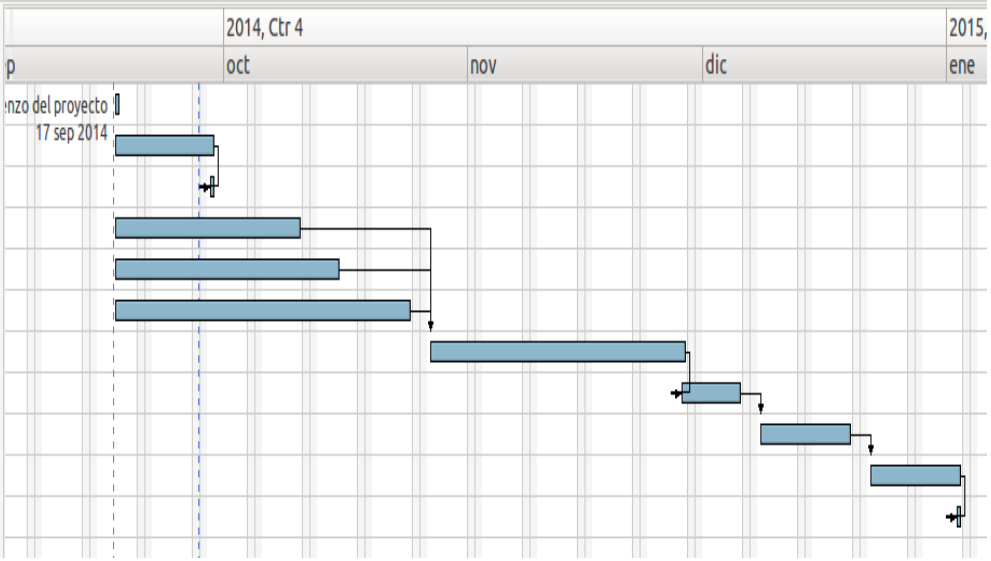

Tabla 4: Diagrama de Gantt

8. Conocimientos Aplicados Adquiridos en el Máster de Software Libre

Los conocimientos adquiridos en el Máster de Software Libre y que fueron aplicados al desarrollo del plug-in de procesado visual con OpenCV para OpenDomo OS llamado OpenCVODOS son múltiples, ya que cada asignatura contribuye a la formación para en conjunto poder desarrollar proyectos como este, pero se destacan las siguientes asignaturas:

- Administración de sistemas GNU/Linux, donde se destaca el uso y programación de shell scripts que permitieron entender y programar los scripts usados en el desarrollo del plug-in.
- Administración avanzada del sistema operativo GNU/Linux, esta asignatura permite comprender más a fondo la estructura de un sistema operativo GNU/Linux y como trabaja el Kernel, esto permitió en el proyecto poder evaluar mejor las librerías de OpenCV necesarias y la forma como se podían integrar al sistema operativo OpenDomo OS.
- En general las asignaturas de desarrollo de aplicaciones que aunque no trabajan con Python, si ayudan a entender otros lenguajes y de esta forma poder aplicar esos conocimientos de una manera más sencilla en el desarrollo de aplicaciones en la industria.

9. Conocimientos Adquiridos en el TFM

Los conocimientos adquiridos en el TFM fueron múltiples, sobretodo de en cuanto a la metodología para en desarrollo de proyectos de Software Libre en ámbitos empresariales, fue una experiencia muy enriquecedora ya que nunca había participado en un proyecto de este tipo, donde el aprendizaje es constante, algunos de los conocimientos adquiridos en el TFM más relevantes son:

- Metodología para desarrollo de aplicaciones reales en Software Libre.
- Uso del repositorio GitHub en entornos reales de desarrollo.
- La forma, estilo de desarrollo y adecuación de aplicaciones por parte de OpenDomo Services S.L, donde con soluciones sencillas pero ingeniosas, realizan cosas muy interesantes a nivel de programación y uso eficiente de GNU/Linux, además son proyectos muy bien delimitados, muy bien definidos y denota que la empresa tiene claro para donde va.

10. Trabajos Futuros

En este trabajo se desarrollo un nuevo plug-in que muy seguramente esta atado a cambios en el tiempo para optimizar su funcionamiento, por ello se han detectado a lo largo del proceso algunas modificaciones a futuro que por el corto tiempo no fueron consideradas en este proyecto, estas se describen a continuación:

- OpenDomo Services S.L. puede integrar el plug-in de procesado visual con OpenCV - OpenCVODOS y sus filtros al plug-in de procesamiento visual existente llamado opendomo-vision, para ser usado vía Web por medio de la interfaz gráfica de administración de su próxima versión OpenDomo OS 2.2
- Sería importante adecuar OpenDomo OS, para que administre el espacio necesario en la instalación de las librerías de OpenCV, ya que estas ocupan 308 MB aproximadamente y este espacio no siempre esta disponible en el sistema operativo usado.
- Por la estructura del plug-in se pueden adaptar nuevos filtros según las necesidades como barreras virtuales, detección de objetos definidos previamente, reconocimiento facial entre otros, para ello se requiere un conocimiento medio o alto en Python, con lo que nacen nuevos proyectos buscando el crecimiento de los servicios que se pueden prestar desde OpenDomo OS.
- Se pueden crear estampillas de tiempo en las imágenes generadas para que no se sobre escriban las imágenes resultantes, esto no se implemento porque no hace parte del plug-in como tal, sino de los filtros en Python, pero se puede considerar a futuro en los filtros.
- Se pueden crear nuevos scripts de instalación de filtros más especializados según los parámetros que sean necesarios, ya que por la variedad de filtros que se pueden crear en Python se hace necesario un proceso de instalación particular para cada uno de ellos.

11.Referencias

[1] OpenDomo Services S.L. “Acerca de” [artículo en línea]. [Fecha de consulta: 15 de octubre de 2014]. <http://es.opendomo.org/about/>

[2] OpenDomo Services S.L. “Probar OpenDomo” [artículo en línea]. [Fecha de consulta: 18 de septiembre de 2014]. <http://www.opendomo.com/wiki/index.php?title=Probar_OpenDomo>

[3] OpenDomo Services S.L. “Sub-proyectos OpenDomo” [artículo en línea]. [Fecha de consulta: 23 de septiembre de 2014]. <https://github.com/search?q=opendomo&ref=cmdform>

[4] Asociación Española de Domótica e Inmótica - CEDOM. “Qué es Domótica” [artículo en línea]. [Fecha de consulta: 10 de septiembre de 2014]. <http://www.cedom.es/sobre-domotica/que-es-domotica>

[5] Axis Communications AB. “Camara de red AXIS 211M” [artículo en línea]. [Fecha de consulta: 17 de octubre de 2014]. <http://www.axis.com/es/products/cam_211m/>

[6] OpenCV. “Open Source Computer Vision Library” [artículo en línea]. [Fecha de consulta: 17 de septiembre de 2014]. <http://opencv.org/>

[7] Bradski G.; Kaehler A. (2008). “Introduction to OpenCV ”, “Getting to Know OpenCV ” y “HighGUI”. En Learning OpenCV [Aprender OpenCV] (Primera edición). Sebastopol, CA, USA:O’Reilly Media, Inc.

[8] Python "The official home of the Python Programming Language" [artículo en línea]. [Fecha de consulta: 17 de septiembre de 2014]. <https://www.python.org/>

[9] OpenCV 3.0.0-dev documentation “Introduction to OpenCV-Python Tutorials” [artículo en línea]. [Fecha de consulta: 18 de septiembre de 2014]. <http://docs.opencv.org/trunk/doc/py_tutorials/py_setup/py_intro/py_intro.html>

[10] Howse J. (2013). "Filtering Images". En OpenCV Computer Vision with Python [OpenCV Visión por computador con Python] (Primera edición). Birmingham B3 2PB, UK:Packt Publishing Ltd.

[11] Debian -- Details of package libopencv-dev in wheezy. “Package: libopencv-dev (2.3.1-11)” [artículo en línea]. [Fecha de consulta: 11 de septiembre de 2014]. <https://packages.debian.org/wheezy/libopencv-dev>

[12] Debian -- Details of package python-opencv in wheezy. “Package: python-opencv (2.3.1-11)” [artículo en línea]. [Fecha de consulta: 11 de septiembre de 2014]. <https://packages.debian.org/wheezy/python-opencv>

[13] Solem J. (2012). "Local Image Descriptors" y "OpenCV". En Programming Computer Vision with Python [Programación de Visión por Computador con Python] (Primera edición). Sebastopol, CA, USA:O’Reilly Media, Inc.

[14] Itseez/opencv. "OpenCV: Open Source Computer Vision Library" [Repositorio de Ejemplos OpenCV en línea]. [Fecha de consulta: 20 de septiembre de 2014]. <https://github.com/Itseez/opencv>

12. Anexos

Anexo 1 GNU GENERAL PUBLIC LICENSE V3

Esta es la licencia que usa OpenDomo OS y el plug-in desarrollado en el proyecto, a continuación se hace copia del texto sin modificación alguna:

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Everyone is permitted to copy and distribute verbatim copies of this license document, but changing it is not allowed.

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps: (1) assert copyright on the software, and (2) offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there

is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

“This License” refers to version 3 of the GNU General Public License.

“Copyright” also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

“The Program” refers to any copyrightable work licensed under this License. Each licensee is addressed as “you”. “Licensees” and “recipients” may be individuals or organizations.

To “modify” a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a “modified version” of the earlier work or a work “based on” the earlier work.

A “covered work” means either the unmodified Program or a work based on the Program.

To “propagate” a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To “convey” a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer

network, with no transfer of a copy, is not conveying.

An interactive user interface displays “Appropriate Legal Notices” to the extent that it includes a convenient and prominently visible feature that (1) displays an appropriate copyright notice, and (2) tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License. If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The “source code” for a work means the preferred form of the work for making modifications to it. “Object code” means any non-source form of a work.

A “Standard Interface” means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The “System Libraries” of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A “Major Component”, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The “Corresponding Source” for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified

Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf, under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

a) The work must carry prominent notices stating that you modified it, and giving

a relevant date.

b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to “keep intact all notices”.

c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.

d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an “aggregate” if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.

b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either (1) a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or (2) access to copy the Corresponding Source from a network server at no charge.

c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.

d) Convey the object code by offering access from a designated place (gratis or

for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server (operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.

e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A “User Product” is either (1) a “consumer product”, which means any tangible personal property which is normally used for personal, family, or household purposes, or (2) anything designed or sold for incorporation into a dwelling. In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, “normally used” refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

“Installation Information” for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product

in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

“Additional permissions” are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or

b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or

c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or

d) Limiting the use for publicity purposes of names of licensors or authors of the material; or

e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or

f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered “further restrictions” within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated (a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and (b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An “entity transaction” is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit) alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A “contributor” is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's “contributor version”.

A contributor's “essential patent claims” are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, “control” includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a “patent license” is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To “grant” such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network

server or other readily accessible means, then you must either (1) cause the Corresponding Source to be so available, or (2) arrange to deprive yourself of the benefit of the patent license for this particular work, or (3) arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients. “Knowingly relying” means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is “discriminatory” if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license (a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or (b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU

Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License “or any later version” applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM “AS IS” WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO

LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the “copyright” line and a pointer to where the full notice is found.

<one line to give the program's name and a brief idea of what it does.>

Copyright (C) <year> <name of author>

This program is free software: you can redistribute it and/or modify

it under the terms of the GNU General Public License as published by

the Free Software Foundation, either version 3 of the License, or

(at your option) any later version.

This program is distributed in the hope that it will be useful,

but WITHOUT ANY WARRANTY; without even the implied warranty of

MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE. See the

GNU General Public License for more details.

You should have received a copy of the GNU General Public License

along with this program. If not, see <http://www.gnu.org/licenses/>.

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

<program> Copyright (C) <year> <name of author>

This program comes with ABSOLUTELY NO WARRANTY; for details type

`show w'.

This is free software, and you are welcome to redistribute it

under certain conditions; type `show c' for details.

The hypothetical commands `show w' and `show c' should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an “about box”.

You should also get your employer (if you work as a programmer) or school, if any, to sign a “copyright disclaimer” for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see <http://www.gnu.org/licenses/>.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read <http://www.gnu.org/philosophy/why-not-lgpl.html>.