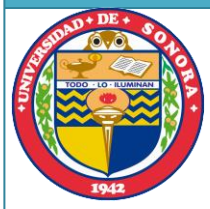


Control Remoto de Robots de Manufactura

REPORTE DE PRÁCTICAS PROFESIONALES

Universidad de Sonora

Alumno:
Barraza Celaya Néstor de Jesús

DICIEMBRE 2013

MARZO 2014

índice

Introducción

La educación impartida en las universidades hoy en día es muy diferente a la educación recibida por nuestros padres o abuelos en donde se centraba en la cátedra que impartía el maestro dentro de un aula, en las disciplinas técnicas y científicas es fundamental complementar el conocimiento teórico con el experimento práctico adquirido en los laboratorios que confirma la teoría.

Cada vez es más frecuente el uso de robots en las grandes industrias, por ello es que cada vez se debe capacitar mejor al alumno para poder manipularlos. En muchas ocasiones la capacitación es muy tardía puesto que debe realizar la manipulación del robo industrial presencialmente.

Con el paso del tiempo, los robots han venido mejorando la calidad en los productos industriales, así como mejorando los tiempos en la elaboración y reduciendo los costos por cuestiones de actividad humana. Para muchas empresas el simple hecho de contar con un asistente robótico en la producción puede significar obtener mayores ganancias, mejorar su posición o simplemente permanecer en el mercado.

También es importante señalar que en cuestiones de educación, la enseñanza de robótica puede ser crucial para que una escuela (llámese universidad en este caso) tenga una mejor opción de colocar a los alumnos en mejores puestos de trabajo.

Las grandes industrias requieren de nuevas técnicas para poder realizar sus actividades, claro está. Pero también es necesario que estas técnicas sean de costo considerable para la empresa y sobre todo que signifiquen una mejora. Es por ello que se pueden combinar algunas actividades para ello: la robótica en su esencia pura y los sistemas de información.

Descripción del Proyecto

En primer lugar, cabe destacar que el proyecto tiene dos enfoques para desarrollarse, uno es el enfoque de la carrera concerniente a Ingeniería mecatrónica y el otro, que es en el que se centra esta parte que será descrita del proyecto: La de Ingeniería en Sistemas de Información. El segundo punto importante es que este es un sistema en conjunto, con la finalidad de dar cátedra a los alumnos, no el de un uso complejo industrial.

Las actividades consisten en crear un sistema para acceder remotamente a un robot pequeño, elaborado por alumnos de la carrera de mecatrónica. Ese robot será controlado por un teach pendant que es un aparato parecido a una calculadora científica que puede controlar a un robot Fanuc remotamente., el cual a su vez será controlado por un acceso remoto mediante el uso de la placa Arduino y el lenguaje de programación para comunicación Python.

Con Python se pretende la realización de la comunicación remota a través de un VNC ya que, cabe destacar que este proyecto será elaborado con software libre, por lo que también es de gran importancia remarcar que se elaborará en sistema operativo Ubuntu. Por lo cual todo el sistema y herramientas serán de uso gratuito con el fin de mejorar y apoyar a la enseñanza del alumno en el manejo de estos aparatos.

Otro de los puntos es que, se debe realizar un sistema en el que se registren los tiempos de uso de un robot por ejemplo, para no crear caos. Para ellos se utiliza PHP, lenguaje que permite el uso de servidores web como Apache. Se hará una combinación con MySQL para el ordenamiento de los datos.

Cuando se entre al sistema, se debe registrar una fecha que ningún otro usuario podrá apartar. Solamente en ese tiempo establecido se podrá establecer la conexión con el sistema (teach pendant elaborado en Python para el control de la placa Arduino) y poder hacer control del robot.

El teach pendant elaborado en Python controlará a un teach pendant físico el cual podrá manipular un brazo de manufactura. Este será controlado por un manipulador con servomotores, controlados a su vez por la placa Arduino. Este será desarrollado por otro alumno de la carrera de Mecatrónica. Además este será monitoreado por una cámara web IP.

Problemática a Resolver

Una de las problemáticas a resolver es la elaboración de un sistema de bajo costo para la escuela, en este caso es importante reducirlo lo más posible, ya sea en cuestión de licencias o de otros dispositivos para su desarrollo.

Hablando puramente del sistema se necesita establecer una conexión segura y confiable para el usuario, que tenga redundancia en caso de que colapse el sistema ya que puede perderse información o no llevarse a cabo la actividad de enseñanza por causas de fallo del sistema.

Se sabe de diversos métodos de comunicación remota, pero en este caso se requiere uno adaptado a medida, por lo cual se requiere de la manipulación de protocolo VNC. Además de elaborar el dispositivo de control teach pendant en un lenguaje compatible con la placa Arduino.

Justificación

Se ha tomado la iniciativa de crear un asistente para acceder y controlar remotamente un robot con el fin de enseñar al alumno de la carrera de mecatrónica los conceptos y operaciones básicas que estos desarrollan en las empresas. Se requiere que todos puedan visualizar los mismos conceptos, cosa que por espacio de las aulas o el tiempo que se necesita el conocimiento no es captado de la misma forma.

Alcances y Limitaciones

Alcances

El sistema funcionando completamente permitirá al alumno manipular un robot con el fin de aprendizaje, sin necesidad de hacerlo presencialmente, lo que elimina la necesidad de estar en un espacio quizás reducido, además de contar con un tiempo establecido y mejor organizado. Además se tiene en consideración que los gastos de desarrollo son mínimos.

Limitaciones

- El uso nulo del lenguaje de programación Python.
- Desconocimiento del uso de la placa Arduino.
- Poca interacción con aplicaciones remotas (programas).
- Uso casi nulo de sistema operativo Linux.
- Redacción.

Objetivos de Proyecto

- Realiza un sistema de control y acceso remoto.
- Manejar los dispositivos usando el lenguaje Pyhton
- Utilizar software libre.

Marco Teórico

Python es un lenguaje de programación interpretado cuya filosofía hace hincapié en una sintaxis muy limpia y que favorezca un código legible. Se trata de un lenguaje de programación multiparadigma, ya que soporta orientación a objetos, programación imperativa y, en menor medida, programación funcional. Es un lenguaje interpretado, usa tipado dinámico y es multiplataforma. Es administrado por la Python Software Foundation. Posee una licencia de código abierto, denominada Python Software Foundation License,[1] que es compatible con la Licencia pública general de GNU a partir de la versión 2.1.1, e incompatible en ciertas versiones anteriores.

VNC es un programa de software libre basado en una estructura cliente-servidor el cual permite tomar el control del ordenador servidor remotamente a través de un ordenador cliente. También llamado software de escritorio remoto. VNC no impone restricciones en el sistema operativo del ordenador servidor con respecto al del cliente: es posible compartir la pantalla de una máquina con cualquier sistema operativo que soporte VNC conectándose desde otro ordenador o dispositivo que disponga de un cliente VNC portado. La versión original del VNC se desarrolló en Reino Unido, concretamente en los laboratorios AT&T Olivetti Research Laboratory, en Cambridge, Reino Unido. El programa era de código abierto por lo que cualquiera podía modificarlo y existen hoy en día varios programas para el mismo uso. Muchos derivados modernos de él son software libre bajo licencia GNU General Public License.

PHP es un lenguaje de programación de uso general de código del lado del servidor originalmente diseñado para el desarrollo web de contenido dinámico. Fue uno de los primeros lenguajes de programación del lado del servidor que se podían incorporar directamente en el documento HTML en lugar de llamar a un archivo externo que procese los datos. El código es interpretado por un servidor web con un módulo de procesador de PHP que genera la página Web resultante. PHP ha evolucionado por lo que ahora incluye también una interfaz de línea de comandos que puede ser usada en aplicaciones gráficas independientes. PHP puede ser usado en la mayoría de los servidores web al igual que en casi todos los sistemas operativos y plataformas sin ningún costo.

MySQL es un sistema de gestión de bases de datos relacional, multihilo y multiusuario con más de seis millones de instalaciones.[1] MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems y ésta a su vez de Oracle Corporation desde abril de 2009— desarrolla MySQL como software libre en un esquema de licenciamiento dual. Por un lado se ofrece bajo la GNU GPL para cualquier uso compatible con esta licencia, pero para aquellas empresas que quieran incorporarlo en productos privativos deben comprar a la empresa una licencia específica que les permita este uso.

Apache es un servidor web HTTP de código abierto, para plataformas Unix (BSD, GNU/Linux, etc.), Microsoft Windows, Macintosh y otras, que implementa el protocolo HTTP/1.1 y la noción de sitio virtual.

Arduino es una plataforma de hardware libre, basada en una placa con un microcontrolador y un entorno de desarrollo, diseñada para facilitar el uso de la electrónica en proyectos multidisciplinares. El hardware consiste en una placa con un microcontrolador Atmel AVR y puertos de entrada/salida. Los microcontroladores más usados son el Atmega168, Atmega328, Atmega1280, ATmega8 por su sencillez y bajo coste que permiten el desarrollo de múltiples diseños. Por otro lado el software consiste en un entorno de desarrollo que implementa el lenguaje de programación Processing/Wiring y el cargador de arranque (*boot loader*) que corre en la placa.

Ubuntu es un sistema operativo basado en Linux y que se distribuye como software libre, el cual incluye su propio entorno de escritorio denominado Unity. Su nombre proviene de la ética ubuntu, en la que se habla de la existencia de uno mismo como cooperación de los demás. Está orientado al usuario novato y promedio, con un fuerte enfoque en la facilidad de uso y en mejorar la experiencia de usuario. Está compuesto de múltiple software normalmente distribuido bajo una licencia libre o de código abierto.

Seguridad y accesibilidad

El sistema incluye funciones avanzadas de seguridad y entre sus políticas se encuentra el no activar, de forma predeterminada, procesos latentes al momento de instalarse. Por eso mismo, no hay un cortafuegos predeterminado, ya que supuestamente no existen servicios que puedan atentar a la seguridad del sistema. Para labores o tareas administrativas en la línea de comandos incluye una herramienta llamada sudo (de las siglas en inglés de **S**witch**U**ser **do**), con la que se evita el uso del usuario administrador. Posee accesibilidad e internacionalización, de modo que el sistema esté disponible para tanta gente como sea posible. Desde la versión 5.04, se utiliza UTF-8 como codificación de caracteres predeterminado.

No sólo se relaciona con Debian por el uso del mismo formato de paquetes .deb. También tiene uniones con esa comunidad, aunque raramente contribuyendo con cualquier cambio directa e inmediatamente, o sólo anunciándolos. Esto sucede en los tiempos de lanzamiento. La maryoría de los empaquetadores de Debian son los que realizan también la mayoría de los paquetes *importantes* de Ubuntu.

Pygame es un conjunto de módulos del lenguaje Python que permiten la creación de videojuegos en dos dimensiones de una manera sencilla. Está orientado al manejo de sprites. Cabe destacar que se utilizará la versión 2.7 por cuestión compatibilidad.

Gracias al lenguaje, se puede prototipar y desarrollar rápidamente. Esto se puede comprobar en las competiciones que se disputan online, donde es cada vez más usado. Los resultados pueden llegar a ser profesionales.

PySerial es una librería de python que permite comunicarse a través de comunicaciones por serial (RS-232). Esto puede ser muy útil para mandar o recibir datos de periféricos que sean comunes o que tú mismo hayas hecho de una manera increíblemente sencilla y sin tener que complicarse para nada con este tipo de programación. [6]

wxPython es una suite de librerías de interfaces gráficas para Python (programadas en C++), destinadas a crear GUIs de forma simple. Funciona como un módulo más de Python, debiendo ser importado al igual que cualquier otro. [4]

Qt Designer. Es un programa (parte del conjuntos de programas para el desarrollo de aplicaciones para el Framework Qt) para desarrollar interfaces gráficas de usuario (multilenguages debido a que genera un archivo XML cuyo contenido es el formato de dicho GUI, pudiéndolo convertir con los programas pertinentes a cada lenguage).

LAMP es el acrónimo usado para describir un sistema de infraestructura de internet que usa las siguientes herramientas:

- Linux, el sistema operativo; En algunos casos también se refiere a LDAP.
- Apache, el servidor web;
- MySQL/MariaDB, el gestor de bases de datos;
- Perl, PHP, o Python, los lenguajes de programación.

La combinación de estas tecnologías es usada principalmente para definir la infraestructura de un servidor web, utilizando un paradigma de programación para el desarrollo.

Twisted es un framework de red para programación dirigida por eventos escrito en Python y licenciado bajo la licencia MIT. Twisted proporciona soporte para varias arquitecturas (TCP, UDP, SSL/TLS, IP Multicast, Unix domain sockets), un gran número de protocolos (incluidos HTTP, XMPP, NNTP, IMAP, SSH, IRC, FTP), y mucho mas. Twisted se basa en el paradigma de la *programación dirigida por eventos*, quiere decir que los usuarios de Twisted pueden escribir pequeños callbacks (retrollamadas) predefinidos en el framework para realizar tareas complejas.

Servomotor (también llamado **servo**) es un dispositivo similar a un motor de corriente continua que tiene la capacidad de ubicarse en cualquier posición dentro de su rango de operación, y mantenerse estable en dicha posición. Un servomotor es un motor eléctrico que puede ser controlado tanto en velocidad como en posición.

Zope es un proyecto comunitario activista de un entorno de desarrollo para la creación de sitios web dinámicos y/o aplicaciones web usando un servidor de aplicaciones web orientado al objeto, escrito en el lenguaje de programación Python *(con algunos componentes fueron escritos en lenguaje C para optimizar su rendimiento)* de código abierto publicado bajo la licencia Zope Public License. Muchas cosas que son actualmente parte del núcleo de Python originalmente fueron innovaciones en su momento dadas por el desarrollo de Zope a la comunidad de desarrolladores Python, un ejemplo de esto es la librería *datetime* que proviene del *DateTime* de Zope 2. [8]

Actividades Desarrolladas

Primeramente, platiqué con el profesor Rafael Castillo sobre el proyecto, puesto que éste parte de una tesis de doctorado que él propuso, por lo tanto el primer paso fue aclarar la idea de lo que buscaba. Una vez empezando estas pláticas quedó más claro la finalidad del proyecto.

Para empezar con el proyecto en esencia, decidí realizar una investigación sobre algunos sistemas que tuvieran las características descritas, tal es el caso de control y acceso remoto, administración de dispositivos, etc. La primera opción sugerida por el maestro fue SCADA, acrónimo de Supervisory Control And Data Acquisition (Supervisión, Control y Adquisición de Datos) es un software para microcomputadoras que permite controlar y supervisar procesos industriales a distancia. Facilita retroalimentación en tiempo real con los dispositivos de campo (sensores y actuadores) y controlando el proceso automáticamente. Provee de toda la información que se genera en el proceso productivo (supervisión, control calidad, control de producción, almacenamiento de datos, etc.) y permite su gestión e intervención. [1]

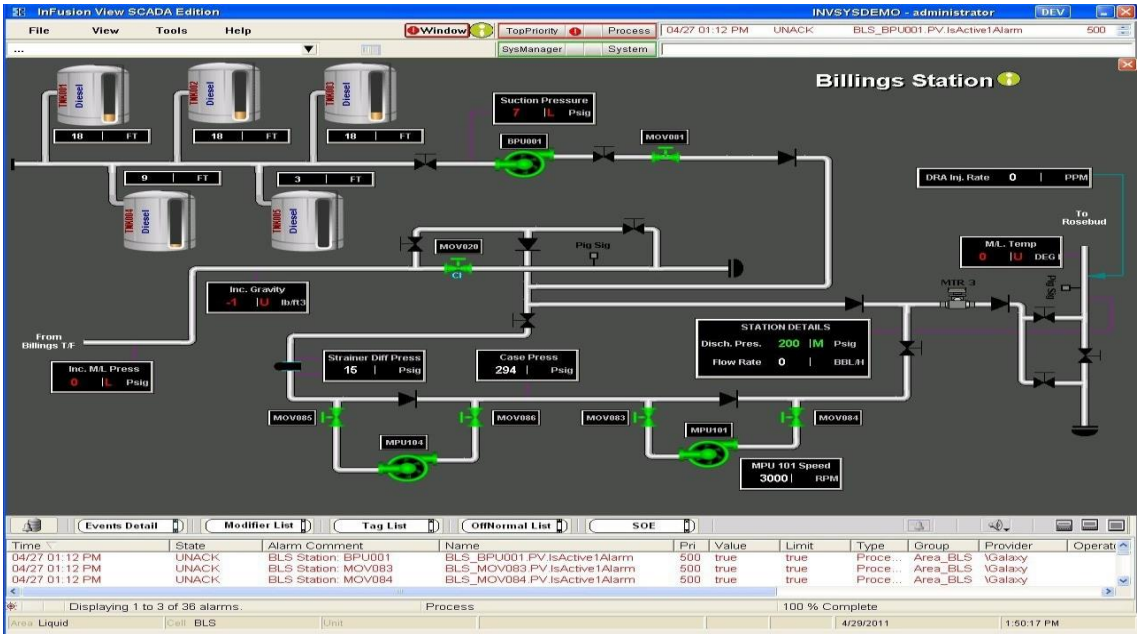
Pero al parecer uno de los inconvenientes es que la curva de aprendizaje es un poco alta, ya que se necesita capacitación a fondo para operarlo, lo cual cuesta. Además de la propia instalación y servicio, con lo cual no va alineado con lo que se busca, la facilidad y un bajo costo. [2]

Operabilidad de SCADA.

Además se pudo encontrar otros tantos como se muestra a continuación con sus respectivos fabricantes:

Nombre de software	Fabricante
Aimax	Desin Instruments S. A.
CUBE	Orsi España S. A.
FIX	Intellution.
Lookout	National Instruments.
Monitor Pro	Schneider Electric.
Scada InTouch	LOGITEK.
SYSMAC SCS	Omron.
Scatt Graph 5000	ABB
WinCC	Siemens.
Coros LS-B/Win	Siemens.
CIRNET	CIRCUTOR S.A.
FIXDMACS	Omron-Intellution.
RS-VIEW32	Rockwell
GENESIS32	Iconics

Algunos de ellos tiene opciones para descargarse, pero la mayoría como opciones de muestra o temporales por lo que tampoco cumplirían con el cometido, agregándole con ello que no todos pueden adaptarse al proceso que se requiere.



Ejemplo de pantalla operable InFusion View SCADA

Cuando se obtuvo la idea general, el siguiente paso fue analizar las herramientas con las cuales podemos elaborar el sistema. Se realizó un análisis en el que se tomó en cuenta que si se requiere de un acceso remoto, se necesita estar conectado en una red. Para ello puede elegirse varios lenguajes de programación. Más sin embargo si se requiere de una conexión externa o de Internet, es necesario un lenguaje de desarrollo web, obviamente. Por ello se decidió elaborar la primera etapa del proyecto en PHP.

Esta primera etapa consiste en el registro y login del cliente, es decir, introducir un usuario y contraseña para poder acceder al sistema. Estos datos serán guardados en una base de datos, utilizando MySQL como gestor de base datos. Con lo cual ya estamos dando forma a nuestra arquitectura LAMP, al también utilizar Apache como nuestro servidor donde albergaremos la información.

Para esta etapa se tenía contemplada una pantalla como la siguiente:

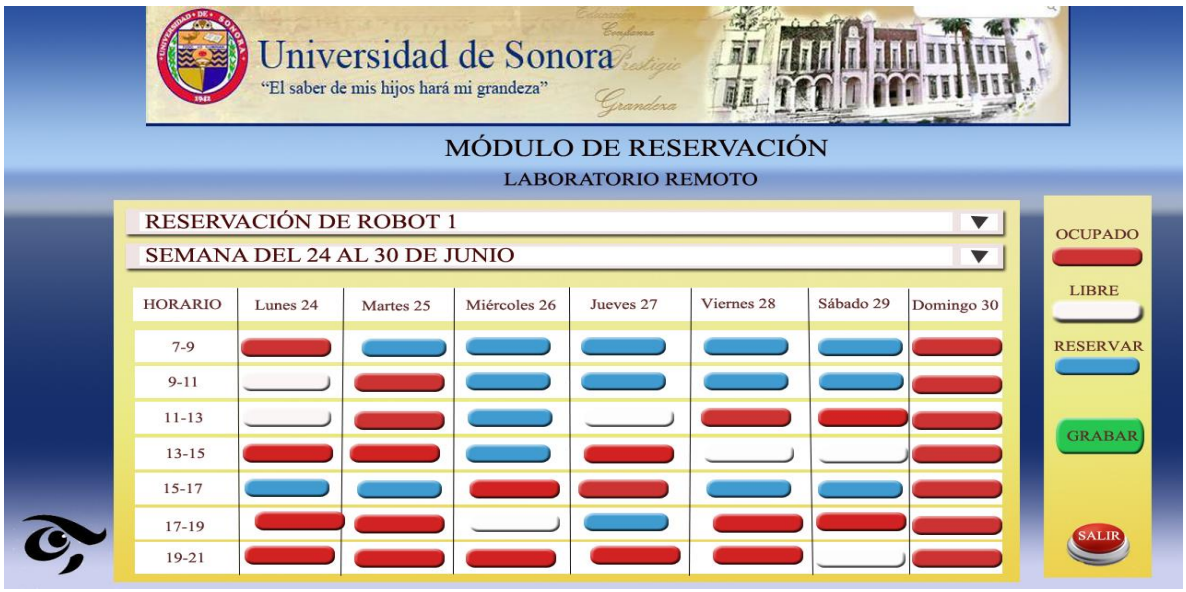
Ejemplo de primera pantalla del sistema

Dicha imagen podemos apreciar que solicitan al usuario un "Usuario" y una "Contraseña" ya otorgadas por el administrador del servicio.

Después pasa a decidir si ya tiene un lugar apartado en el tiempo de uso del laboratorio remoto, para ello se pensó en una pantalla como la siguiente:

Pantalla de reservación de tiempo

En caso de ya tener un tiempo asignado, el sistema comprobará que el usuario este correctamente establecido en fecha y hora de uso del laboratorio y con ello hará la conexión. En caso de no ser así pasará a la siguiente pantalla.



Elección de fecha y hora para reservar

En esa parte, el usuario selecciona un día y hora tal, de una semana "x" de un mes "y" donde los espacios en blanco ("Libre") lo permitan. Si hay un espacio marcado con rojo ("Ocupado") significa que alguien más tiene ese lugar apartado, si está en azul

("Reservar") quiere decir que está próximo a reservarse por el usuario en turno. Para completarla acción se da en "Grabar" o en "Salir" en caso de que no se desee apartar lugar.

Después se planteó otra pantalla donde se puede elegir algún robot. Uno de los disponibles es un FANUC. Luego, ya siendo elegido en este caso FANUC se pasa a una pantalla de control como la que se muestra a continuación.

Pantalla de control

Para este proceso se pensó elaborarse en Windows, ya que PHP es leguaje de programación web, no tendría dificultad de ejecutarse en Linux.

La siguiente etapa resulta un poco más interesante y difícil de desarrollar, puesto que representa el uso de herramientas y lenguajes desconocidos, pues ya implica el uso de Python y de la manera en que se puede establecer las conexiones remotas.

Para las conexiones remotas se tomaron en cuenta algunos programas como por ejemplo Real VNC. El cuál representaba un claro problema porque no se puede controlar (a menor de romper el código fuente) los inicios de sesión como se esperaba, añadiendo que solo permite el uso de la conexión en 5 máquinas diferentes, cosa que quizás después represente un problema. Por ello, esta forma de comunicarse se descartó por completo y decidí buscar alguna herramienta que me permitiera lograr la conexión remota.

Antes de nada, para hacer la conexión, se debe tener una PC con Linux Ubuntu. El proceso de instalación es relativamente sencillo, cambiando un poco por las cuestiones de agregar redes y otras especificaciones. [10]

La primer opción de instalación fue la de Ubuntu 13.04 puesto que en estos momentos es la versión más reciente, pero haciendo pruebas de conexión por VNC dentro del propio Ubuntu tenia errores de conexión, por lo tanto se optó por instalar la versión Ubuntu 12.04 que al parecer es más estable, ya que se hicieron las pruebas de conexión con el comando interno de VNC y no se tuvieron problemas. La versión de VNC es "*tightvnc server*" y "*tightvnc viewer*" probada de un Ubuntu 12.04 a un Kubuntu 12.04. Para realizar la instalación de VNC se utiliza el siguiente comando: **"sudo apt-get -y install ubuntu-desktop tightvncserver"**

Ubuntu Server 12.04

Una vez haciendo la prueba, se pasa a instalar la versión de escritorio de la siguiente manera.

sudo apt-get update
sudo apt-get install ubuntu-desktop

Estos comandos harán el proceso de descarga y de instalación del modo gráfico en el servidor.

Para instalar el soporte de idioma español, abrimos la Terminal y vamos poniendo de uno en uno:

sudo apt-get update
sudo apt-get install language-pack-es

sudo apt-get install language-pack-es-base
sudo apt-get install language-pack-gnome-es
sudo apt-get install language-pack-gnome-es-base
sudo apt-get install language-selector
sudo apt-get install language-support-es

Después de instalar todos los soportes, instalamos gksu para que funcionen correctamente los menús:

sudo apt-get install gksu

Para instalar las Herramientas de red tecleamos esto en la Terminal:

sudo apt-get install gnome-system-tools gnome-nettool

Una vez instalado, y esto es lo más importante, tenemos que llamar a la interfaz por primera vez, para llamar la interfaz escribimos:

startx

Y ya con eso tenemos la versión gráfica para Ubuntu Server.

Volviendo con la cuestión de la conexión, se presentaba otro problema, no se puede administrar adecuadamente las sesiones, puesto que se requiere cada vez entrar con el mismo usuario y contraseña establecidos en la conexión del servidor. Por ello no cumplía con el propósito. Para ello se buscó un VNC que pueda ser manejable desde su código. La solución fue un VNC hecho en Python, el cual puede manipularse para direccionarse a una IP establecida, con una contraseña que reconoce un VNC Server.

Basta con ejecutar el comando
x11vnc –forever –usepw –httpdir /usr/share/vnc-java/ -httpport 5800

Con este comando se ejecuta el servidor previamente instalado y con una contraseña establecida para ello se utiliza el comando siguiente:

xx11vnc –storepasswd

Establecemos la contraseña, la cual se usará también en el código de Python para el vnc viewer.

```
class Options(usage.Options):
    optParameters = [
        ['display',     'd', '0',              'VNC display'],
        ['host',        'h', '148.225.67.64',  'remote hostname'],
        ['outfile',     'o', None,             'Logfile [default: sys.stdout]'],
        ['password',    'p', None,             'VNC password'],
        ['depth',       'D', '32',             'Color depth'],
    ]
    optFlags = [
        ['shared',      's',                   'Request shared session'],
        ['fast',        'f',                   'Fast connection is used'],
    ]
```

En este caso , utilizamos una IP fija (148.225.67.64) la cual corresponde a servidor donde se hará la conexión.

Antes de que pudiera arrancar como se debe, se instalaron las librerías Zope Interfaces, Twisted y Pygame en la PC donde se ejecutará el VNC Viewer. Se instalaron en una PC con sistema operativo Windows de 32 bits.

Se hicieron pruebas de seguridad en la conexión, para verificar las vulnerabilidades. Se utilizó el programa Snort , LOIC y Nmap para dichas pruebas

Snort es un sniffer de paquetes y un detector de intrusos basado en red (se monitoriza todo un dominio de colisión). Es un software muy flexible que ofrece capacidades de almacenamiento de sus bitácoras tanto en archivos de texto como en bases de datos abiertas como lo es MySQL. Implementa un motor de detección de ataques y barrido de puertos que permite registrar, alertar y responder ante cualquier anomalía previamente definida. Así mismo existen herramientas de terceros para mostrar informes en tiempo real (ACID) o para convertirlo en un Sistema Detector y Preventor de Intrusos.

Low Orbit Ion Cannon (abreviado LOIC) es una aplicación diseñada para realizar un ataque de denegación de servicio enviando una gran cantidad de paquetes TCP, paquetes UDP o peticiones HTTP con objeto de determinar cuál es la cantidad de peticiones por segundo que puede resolver la red objetivo antes de dejar de funcionar.

Nmap es un programa de código abierto que sirve para efectuar. Se usa para evaluar la seguridad de sistemas informáticos, así como para descubrir servicios o servidores en una red informática.

El siguiente paso una vez probando la conexión y tener en primera fase del registro, es la de elaborar la aplicación en Python para controlar Arduino. La aplicación necesita una interfaz gráfica para la comprensión y fácil manejo la placa.

Para realizar la interfaz se buscaron las herramientas necesarias para poder elaborarla. Primero indagué en el uso de Boa Constructor que es un IDE para Python y wxPython GUI Builder. Ofrece la creación de marco visual y la manipulación, un inspector de objetos, muchos puntos de vista sobre la fuente como los navegadores de objetos, jerarquías de herencia, cadena de documentación generada documentación HTML, un depurador avanzado y ayuda integrada, pero el detalle es que el techa pendant mencionado anteriormente necesita aproximadamente 62 botones para funcionar, con Boa Constructor solo permitía la colocación de 32 botones.

Teach Pendant

Para ello se buscaron más opciones, algunas más complejas que otras, por ejemplo en el caso de IDE Ninja, que por su instalación un poco engorrosa y su poca documentación (puesto que es de reciente creación) decidí no trabajar con él. La opción de hacer la herramientas visual de manera tradicional (empleando un intérprete de Pyhton en consola) complicaba un poco las cosas por el hecho de la ubicación de los diferentes objetos.

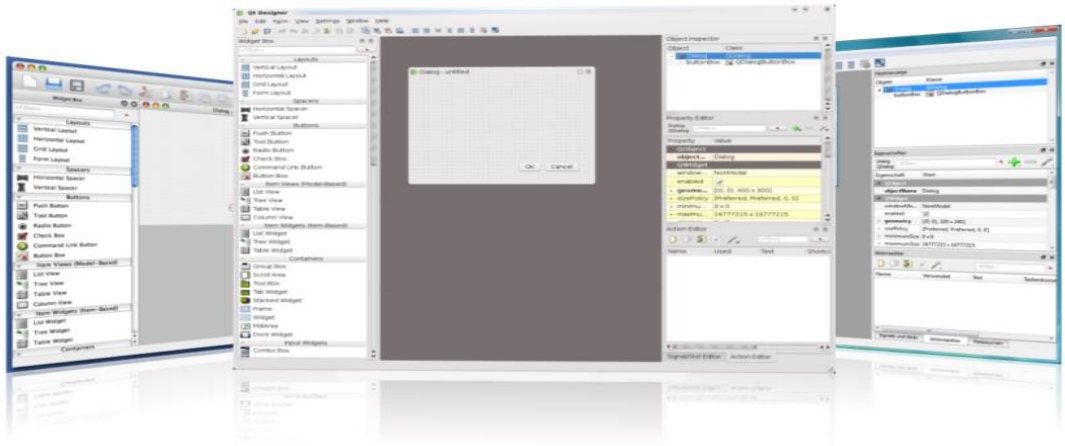
La herramienta adecuada para realizar esta parte fue QT Designer, el cual es una IDE para Python que funciona a perfección en sistemas Linux. Es muy parecido al entorno de desarrollo de Visual Studio ya que se puede simplemente arrastrar la herramienta que se desea colocar.

Con esta herramienta, solo se necesita acomodar los objetos que necesitamos, cambiando o modificando sus atributos y el programa crea el código.
Para realizar la instalación de esta herramienta, en nuestro sistema Ubuntu, abrimos la terminal, y colocamos el siguiente comando:

sudo apt-get install libqt4-core libqt4-dev libqt4-gui

Después se instalan las herramientas que permiten la visualización del código que se usa posteriormente.

sudo apt-get install qt4-dev-tools



Ventana de inicio de QT Designer

Para realizar la conexión entre Python y Arduino se hizo lo que a continuación de describe.
En primer lugar se instaló el IDE para Arduino, el cual puede descargarse desde su página
http://arduino.cc/en/Main/Software, los pasos para instalarse son sencillos, como cualquier programa en Windows, más sin embargo en Ubuntu cambia la manera de instalación. Para instalarlo en Ubuntu debemos seguir lo siguientes pasos.

Abrimos una terminal e instalamos el OpenJDK 7
sudo apt-get install openjdk-7-jdk

Instalar los compiladores de AVR
sudo apt-get install gcc-avr avr-libc

Ir al directorio donde se va instalar el software
Descargar la versión de Arduino 1.0 de acuerdo a la arquitectura del sistema
Para 32 bits
wget http://arduino.googlecode.com/files/arduino-1.0-linux.tgz
Para 64 bits
http://arduino.googlecode.com/files/arduino-1.0-linux64.tgz

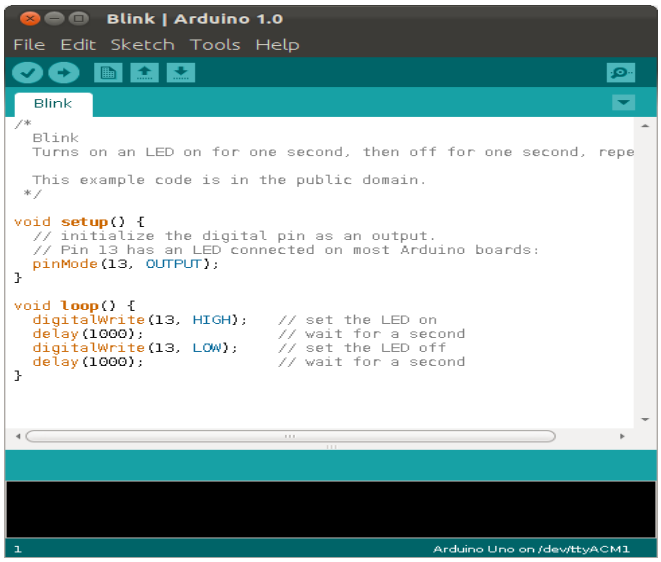
Adicionar el permiso de lectura de puerto USB a su usuario
sudo usermod -a -G dialout $USER

Nos dirigimos a la carpeta donde lo instalamos y tecleamos **./arduino** y con eso lo ejecutamos.

Después con el IDE de Arduino escribimos el siguiente código:

```
int incomingByte = 0; // for incoming serial data
void setup() {
Serial.begin(9600); // opens serial port, sets data rate to 9600 bps
pinMode(13, OUTPUT);
}
void loop() {
int led = 0;
// send data only when you receive data:
if (Serial.available()) {
// read the incoming byte:
incomingByte = Serial.read();
// say what you got:
Serial.print("I received: ");
Serial.println(incomingByte, DEC);
if (incomingByte == 'z') {
digitalWrite(13, true);
}
else {
digitalWrite(13, false);
}
}
}
```

Esa parte del código se encarga de establecer una comunicación con Python, en donde al recibir un carácter "z" el led 13 (por default en la placa Arduino) se enciende y al recibir uno distinto se apaga. Este usa el puerto 9600 apoyados por la librería antes mencionada PySerial. [3]



Pantalla Arduino IDE

Se hizo una pequeña pantalla en Python (modo gráfico) en la cual se creaban dos botones usando la librería WxPython y después probada con QT Designer, en donde uno manda un carácter "z" y otro el carácter "a". [9]

Este es parte del código utilizado:

```
def OnSeven(self, event):
        if self.formula:
            self.display.Clear()
            self.formula = False
        self.display.AppendText('a')
        ser.write('a')

    def OnEight(self, event):
        if self.formula:
            self.display.Clear()
            self.formula = False
        self.display.AppendText('z')
        ser.write('z')
```

Las primeras tres líneas que se muestran a continuación son de gran importancia, ya que es ahí donde se hace la parte de la conexión entre Python y Arduino. En la primera, importamos la librería para hacer la interfaz gráfica, la segunda, decimos que llamamos a la librería de comunicación serial (que contiene PySerial), y en la tercera, le decimos que utilice el puerto COM3 establecido por default en Arduino y que se comunica por el 9600, esto claro no está en la identacion requerida en Python.

```
import wx
import serial,sys, time, psutil
ser = serial.Serial('COM3',9600)
```

Ejemplo de Placa Arduino con el Led 13 (verde) encendido

Otro punto a tratar es sobre el uso de la cámara web. Esta debe ser colocada en un frame. Para las pruebas se utilizó una cámara plug and play True Basix, donde su manejo es más sencillo, puesto que solo se requiere una parte pequeña de código, como se muestra a continuación:

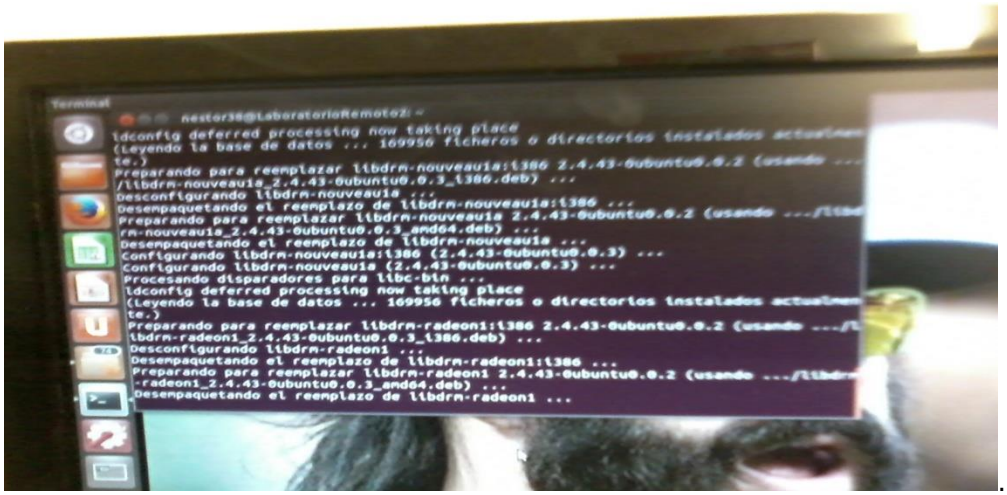
```
import pygame.camera
import pygame.image
import sys

pygame.camera.init()

cameras = pygame.camera.list_cameras()

print "Using camera %s ..." % cameras[0]

webcam = pygame.camera.Camera(cameras[0])

webcam.start()

# grab first frame
img = webcam.get_image()

WIDTH = img.get_width()
HEIGHT = img.get_height()

screen = pygame.display.set_mode( ( WIDTH, HEIGHT ) )
pygame.display.set_caption("pyGame Camera View")

while True :
    for e in pygame.event.get() :
        if e.type == pygame.QUIT :
            sys.exit()

    # draw frame
    screen.blit(img, (0,0))
    pygame.display.flip()
    # grab next frame
    img = webcam.get_image()
```

Solo se importa las librerías de la cámara web que ya está contenida en los módulos instalados de Python. Esta parte fue elaborada tanto en Windows como en Ubuntu, donde en ambos se obtuvo un buen funcionamiento.

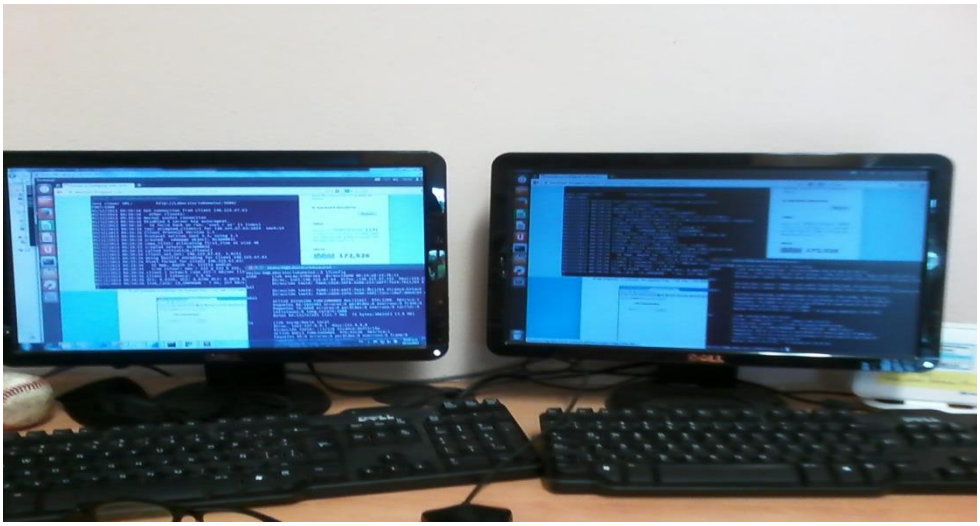
Resultados Obtenidos

Como parte de los resultados obtenidos, podemos mencionar que:

En el caso de la conexión con VNC los resultados fueron satisfactorios en esta etapa, pues se puede manipular el código sin mucha complicación, es entendible y la conexión en si no causa muchos problemas. Lo que sí es necesario decirse es que pueden observar fallos en una conexión muy lenta, el retraso puede ser considerable propiciando un fallo, y es un detalle que necesita refinarse, ya sea teniendo una mejor estructura de red u obteniendo otra fuente de VNC de fácil manejo.



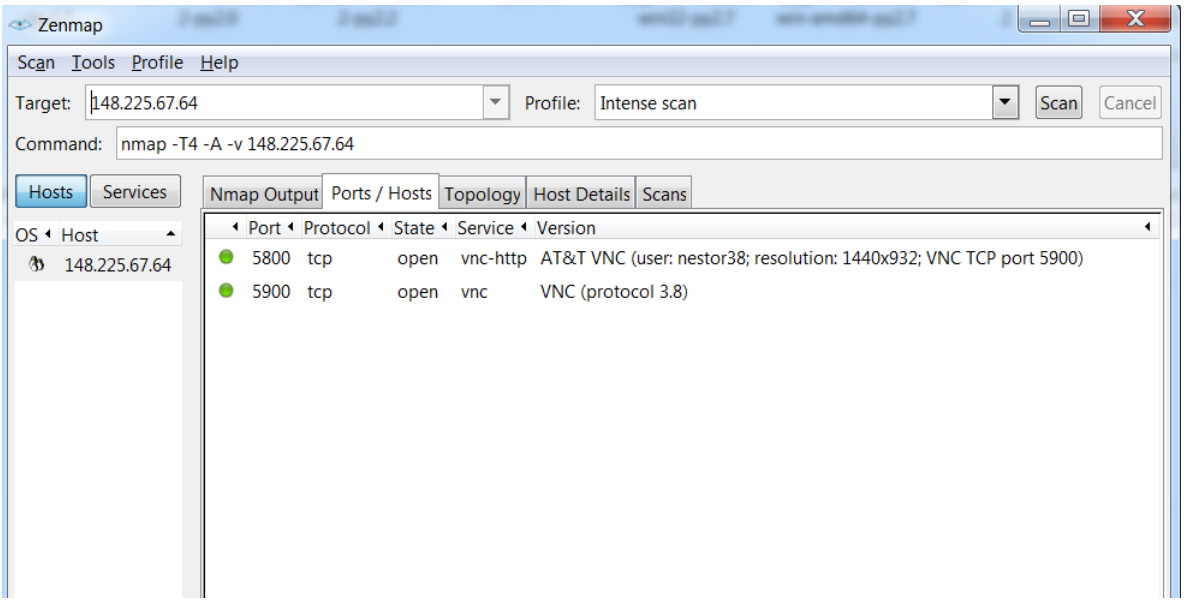
Ejemplo de VNC Server corriendo en Ubuntu



VNV Viewer y VNC Server

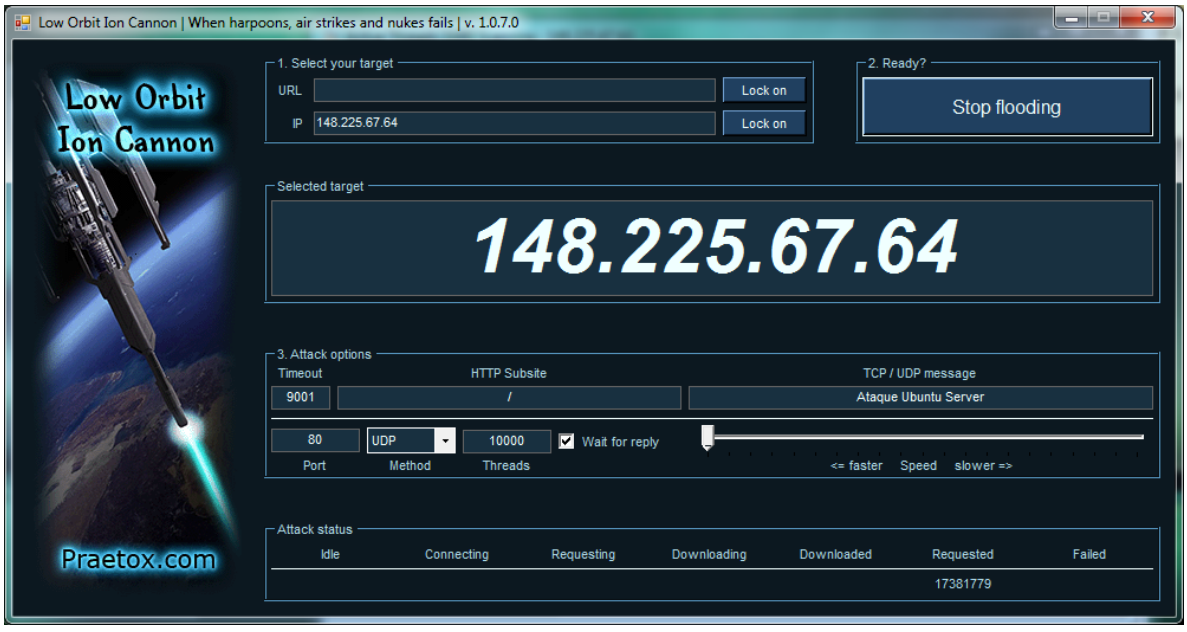
A la izquierda se observa VNC Viewer hecho en Python corriendo desde una PC con Windows 7 de 32 bits y en la derecha se observa el VNC Server en una PC con Ubuntu 12.04 LTS con una arquitectura de 64 bits.

Otro de los resultados fue en cuanto a las observaciones y escaneos de VNC con programas para mejorar la seguridad, como por ejemplo Nmap.

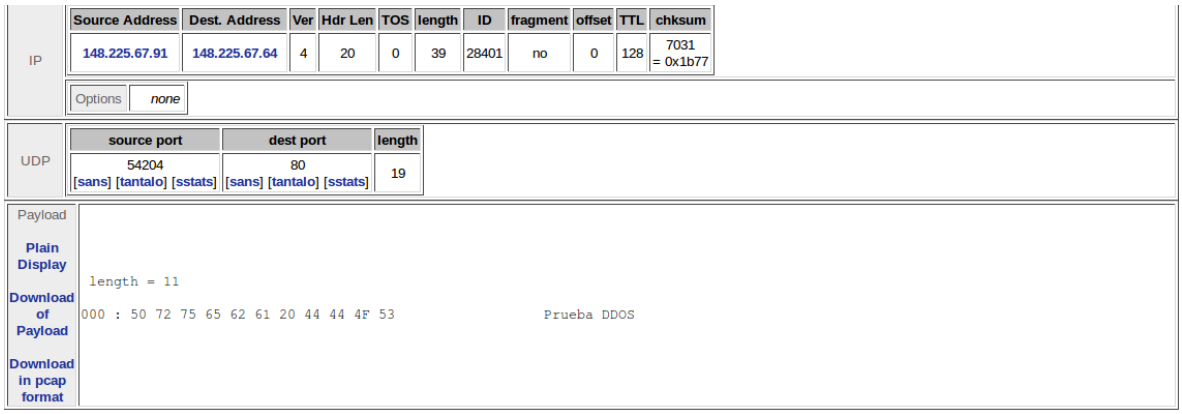


Escaneo con Nmap

Como se muestra en la imagen, los puertos 5800 Y 5900 están vulnerables, por lo que se hizo un ataque con paquetes UDP para ver que tan buena era la conexión de VNC. Para dicha acción se utilizó el programa LOIC.



Para ver si estos paquetes llegaban a su destino, se utilizó el programa Snort, corriendo en Ubuntu.



IP

Source Address	Dest. Address	Ver	Hdr Len	TOS	length	ID	fragment	offset	TTL	chksum
148.225.67.91	148.225.67.64	4	20	0	39	28401	no	0	128	7031 = 0x1b77

Options: none

UDP

source port	dest port	length
54204 [sans] [tantalo] [sstats]	80 [sans] [tantalo] [sstats]	19

Payload

Plain Display

Download of Payload

Download in pcap format

```
length = 11
000 : 50 72 75 65 62 61 20 44 44 4F 53                Prueba DDOS
```

Snort

Con este programa [7] observamos que una máquina atacante (148.225.67.91) manda un paquete a un destino (servidor Ubuntu con dirección 48.225.67.64) con el texto "Prueba DDOS". Por lo tanto es un hecho que le falta seguridad a la conexión, para mejorarla se puede hacer una encriptado de datos (aún no implementado) a través de SSH por lo cual resolvemos problemas de *"man in the middle" y de sniffeo,* usando un túnel ssh en los puertos correspondiente:

ssh -C -o CompressionLevel=9 -f -L 5900:localhost:5900 USUARIO@SERVIDOR_REMOTO sleep 15 ; xvncviewer localhost:5900

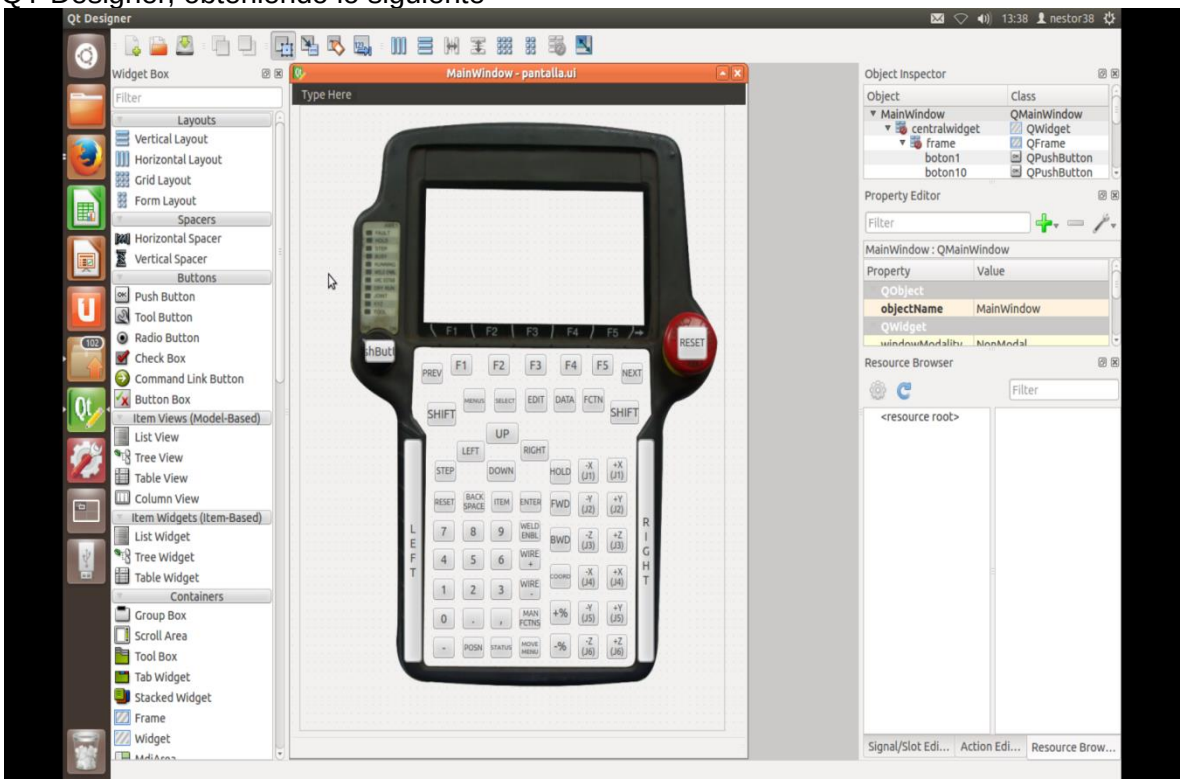
Y también podemos filtrar la información del mismo puerto con iptables y degenamos el resto:

iptables -A INPUT -s VNCVIEWER -p tcp -m tcp –dport 5900 -j ACCEPT
iptables -A INPUT -p tcp -m tcp –dport 5900 -j DROP

En cuanto a la conexión de la cámara web, no hubo problema alguno, solo falta manipularlo para ponerlo todo en la misma ventana, pero eso se hará más adelante.

Cámara web en Ubuntu.

Por otra parte, para la parte fuerte, que es la interfaz gráfica, se hizo primero el diseño en QT Designer, obteniendo lo siguiente



Teach Pendant hecho de forma gráfica en QT Designer

QT Designer nos crea el proyecto con terminación .ui, por lo que se pasa a extensión .py de Python de la siguiente manera:

Nos dirigimos a la carpeta donde guardamos el proyecto, una vez ahí instalamos las dev tools que nos permitirán la conversión para ello usamos el comando:

sudo aptitude install pyaqt4-dev-tools

Después colocados en la carpeta tecleamos:

pyuic4 -x nombredelarchivo.ui -o nombredelarchivo.py

Con esto convertiremos el archivo .ui a .py y lo podemos manipular más fácilmente. Y para ver ahora el código utilizamos el siguiente comando estando dentro de la carpeta del archivo:

sudo vim nombredelarchivo.py

Con esto nos despliega en la terminal todo el código generado y listo para usarse.

```
#      by: PyQt4 UI code generator 4.9.1
#
# WARNING! All changes made in this file will be lost!

from PyQt4 import QtCore, QtGui

try:
    _fromUtf8 = QtCore.QString.fromUtf8
except AttributeError:
    _fromUtf8 = lambda s: s

class Ui_MainWindow(object):
    def setupUi(self, MainWindow):
        MainWindow.setObjectName(_fromUtf8("MainWindow"))
        MainWindow.resize(617, 785)
        self.centralwidget = QtGui.QWidget(MainWindow)
        self.centralwidget.setObjectName(_fromUtf8("centralwidget"))
        self.frame = QtGui.QFrame(self.centralwidget)
        self.frame.setGeometry(QtCore.QRect(10, 0, 579, 731))
        self.frame.setStyleSheet(_fromUtf8("QFrame{\n"
"background-image:url( /home/nestor38/programaspython/caratula4.png);\n"
"}"))
        self.frame.setFrameShape(QtGui.QFrame.StyledPanel)
        self.frame.setFrameShadow(QtGui.QFrame.Raised)
        self.frame.setObjectName(_fromUtf8("frame"))
        self.boton1 = QtGui.QPushButton(self.frame)
        self.boton1.setGeometry(QtCore.QRect(181, 620, 31, 27))
        self.boton1.setObjectName(_fromUtf8("boton1"))
        self.boton2 = QtGui.QPushButton(self.frame)
        self.boton2.setGeometry(QtCore.QRect(221, 620, 31, 27))
        font = QtGui.QFont()
        font.setPointSize(7)
        font.setBold(False)
        font.setItalic(False)
        font.setWeight(50)
```

Ejemplo de código generado

Conclusiones

El proyecto implica el aprender nuevas cosas, conocer nuevos lenguajes, nuevos dispositivos, por ejemplo Python que era desconocido para mí, hoy lo considero como uno de los grandes lenguajes puesto que puede realizarse una gran cantidad de actividades de desarrollo, tiene muchas utilidades y librerías. Pienso que tiene un gran potencial y debería ser enseñado en algunas clases.

Otro punto fue el de manejar otro sistema operativo, conocer las ventajas y desventajas. El software libre sin duda, está hecho para que no tengamos barreras al momento de desarrollar o que se rompan las que ya existen, los servidores son base Linux, por lo tanto, conocer un poco sobre este sistema puede ampliar la visión de sus funciones. El simple hecho de realizar la gran mayoría de las actividades por medio de una terminal, hace que de algún modo aprendas casi sin darte cuenta.

El uso de la placa Arduino me pareció demasiado interesante, aunque no es algo propiamente de la carrera de Ingeniería en Sistemas de Información me gustó el poder manipularlo, ya que ver el resultado de las líneas de código en algo tangible como este hardware resulta fascinante. Así como también el uso de programas para conexiones remotas, las cuales pueden sernos de gran utilidad en nuestras labores.

La cuestión de seguridad también es algo que me agradó aunque fue una actividad bastante corta, pero muy provechosa, puesto que esas actividades generan conocimiento y promueven la investigación.

Y hablando de investigación, es una de las partes que más me ha gustado. Indagar en cómo puedo hacer mis actividades, las herramientas que existen, las búsquedas un poco más allá, es de las actividades que más me agradaron en este proyecto.

Por último, cabe señalar que el proyecto hasta hoy no ha sido concluido puesto que se requiere la coordinación de las dos partes interesadas (Mecatrónica y Sistemas) lo cual se espera conjuntarse y resolverse pronto.

Como nota cabe resaltar que la finalidad de este proyecto es didáctico, mejorando a la enseñanza del alumno en el proceso de aprendizaje de manipulación de robots y dispositivos relacionados.

Recomendaciones

Como una recomendación personal, sería bueno tener materias como "Lenguajes de programación" abiertas como optativas, puesto que se puede dar un panorama general de diversos lenguajes y la utilidad de ellos, así uno puede decidir con que programará en ciertas actividades o proyectos. Además sería más fácil de utilizarlo en un fututo puesto que ya se estaría familiarizado con el lenguaje, que aunque es verdad la frase "desarrollando bien la lógica, el lenguaje es lo de menos" no estaría de más.

Bibliografía

[1] Lorandi. A., Hermida, G., Hernández, J., Ladrón, E. "Los Laboratorios Virtuales y Laboratorios Remotos en la Enseñanza de la Ingeniería“, Revista Internacional de Educación en Ingeniería, Volumen 4, No. 1, 2011

[2] Dormido, S., Sánchez, J., Morilla, F. “Laboratorios virtuales y remotos para la práctica a distancia de la automática”
http://www.dia.uned.es/~fmorilla/Ultimas_publicaciones/2000_LVR_JA00.pdf

[3]http://playground.arduino.cc/Linux/Ubuntu

[4]www.wxpython.org/

[5]http://www.lsi.upc.edu/~lpv/plmaster.dir/charalam/index.html

[6]http://pyserial.sourceforge.net/

[7]www.snort.org

[8]http://docs.zope.org/zope.interface/README.html

[9]playground.arduino.cc/interfacing/python

[10]www.ubuntu.com/