Universidad Autónoma Metropolitana
Unidad Azcapotzalco
División de Ciencias Básicas e Ingeniería
Licenciatura en Ingeniería en Computación

Análisis, diseño e implementación de protocolos de aplicación para sistemas sensibles al contexto

Modalidad: Proyecto de Investigación

Trimestre 2018 Otoño

Ricardo Arturo Martínez Gutiérrez
2132004072
ricardo.amgutierrez3@gmail.com

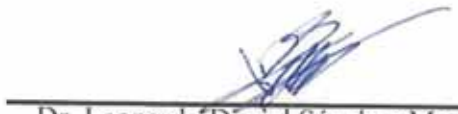
Dr. Leonardo Daniel Sánchez Martínez
Asesor
Departamento de Sistemas
ldsm@correo.azc.uam.mx

Dr. José Alejandro Reyes Ortíz
Coasesor
Departamento de Sistemas
jaro@correo.azc.uam.mx

27 de julio de 2018

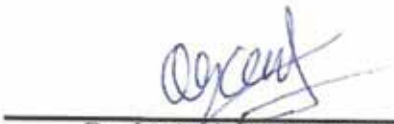
Declaratoria

Yo, Leonardo Daniel Sánchez Martínez, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.



Dr. Leonardo Daniel Sánchez Martínez

Yo, José Alejandro Reyes Ortíz, declaro que aprobé el contenido del presente Reporte de Proyecto de Integración y doy mi autorización para su publicación en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.



Dr. José Alejandro Reyes Ortíz

Yo, Martínez Gutiérrez Ricardo Arturo, doy mi autorización a la Coordinación de Servicios de Información de la Universidad Autónoma Metropolitana, Unidad Azcapotzalco, para publicar el presente documento en la Biblioteca Digital, así como en el Repositorio Institucional de UAM Azcapotzalco.

Ricardo Arturo Martínez Gutiérrez

Resumen

El siguiente proyecto se basa en el desarrollo de un protocolo de aplicación para el almacenamiento de datos generados por una red de sensores en formato XML. El almacenamiento de información se realiza en un Disco Duro Externo para su posterior análisis. Para ello, el proyecto se enfoca en un sistema sensible al contexto que proporcionará información relacionada al ambiente que rodea a un conjunto de usuarios, como temperatura, humedad, luminosidad, etc., la cual se almacenará en un disco duro. Dicha información se clasificó en dos categorías, "importante" y "secundaria". Para ello se desarrolló un programa en Python que divide los datos en 2 carpetas en un disco duro externo. El disco duro contiene el sistema operativo Raspbian funcionando en su respectiva Raspberry y su objetivo principal es operar como nodo central de la red del sistema, teniendo la capacidad de tomar la decisión de a dónde enviar los datos y almacenarlos en una base de datos

Para lograr hacer la lectura necesaria desde cualquier computadora conectada en red, se elaboró un NAS que brinda una mayor facilidad en el compartimiento de archivos.

Finalmente se tiene conectividad remota desde cualquier computadora del sistema y de esta forma dejar 4 HDD funcionales en una Raspberry para tener una gran capacidad de almacenamiento en y toda la transmisión de lleve a cabo en y desde la Raspberry.

Tabla de contenido

Introducción

Los sistemas sensibles al contexto son sistemas que buscan ofrecer servicios a distintos usuarios con base en sus preferencias y en el contexto en el que se encuentran. Se entiende por contexto todo aquello tangible e intangible que rodea al usuario y que impacta en su vida diaria. En el caso de aquello tangible, se utilizan diversas herramientas como sensores que constantemente monitorea el contexto del usuario en espera de un evento. En particular, las redes de sensores generan una gran cantidad de datos que se deben almacenar para su posterior explotación. La cantidad de información generada por este tipo de redes es tan grande que deriva en problemas de almacenamiento e inundación de la red debido a la transmisión y almacenamiento de la misma información. Estos problemas se resuelven agregando mayores capacidades de almacenamiento (de 1TB en adelante), a los dispositivos que conforman la red o bien, se resuelven aplicando criterios de clasificación de información para almacenar y transmitir únicamente aquella información que se considere de interés. En ese sentido, los protocolos de aplicación son la respuesta inmediata a este tipo de problemas, ya que estos dictan la información que resulta interesante para el contexto de un usuario y cuál debe ser o no almacenada.

En particular, el Área de Investigación en Sistemas de Información Inteligentes (AISII) desea monitorear un espacio académico del que desea almacenar toda información generada por la red que está en el mismo. Esta se conforma por distintos nodos:

- Nodo sensor o NS. Recolecta las variables del entorno (temperatura, humedad, luz) y las difunde por una canal WiFi a un Nodo Concentrador.
- Nodo identificador o NI. Identifica o detecta la presencia de personas y las notifica vía WiFi a un Nodo Concentrador.
- Nodo concentrador o NC. Recolecta y almacena toda la información de los NS y NI para posteriormente difundirla a un servidor central.

Todos los nodos que conforman la red son dispositivos mínimos con capacidades limitadas en cuanto almacenamiento y procesamiento se refiere, lo que supone un problema al momento de almacenar y difundir la información. Además, se requiere una clasificación de la información generada por la red para poder determinar la información que resulta prioritaria para su difusión en la red. En este proyecto se propone diseñar un sistema de almacenamiento con un protocolo de aplicación para dividir la información de un sistema sensible al contexto en primaria y secundaria. La idea es almacenar toda la información en carpetas separadas para poder analizarla posteriormente, tomando en cuenta que en el diseño del sistema de almacenamiento se usará un HDD (Unidad de Disco Duro) externo en cada NC.

Antecedentes

Actualmente, existen diversas soluciones que permiten almacenar una gran cantidad de datos, sin embargo, algunos de ellos lo almacenan utilizando formatos enfocados a aplicaciones geoespaciales, como el trabajo presentado en [4].

Se usan los archivos logs de hoy en día, esto para observar la información que se recolecta y poder analizarla, ya sea desde un servidor UNIX o incluso un servidor Windows. Dichos logs

son generados desde el mismo programa que se ejecuta, se puede observar en el trabajo presentado en [7].

El almacenamiento es fundamental para todos, por esto mismo existen diversas soluciones, como el uso del Big Data, una NAS interna, Servidores UNIX, etc., para poder guardar datos importantes o incluso lanzar a producción los aplicativos y dichos aplicativos generan aún más información (dependiendo de que se está ejecutando), así se muestra en el trabajo presentado en [1].

De hoy en día los errores cometidos desde servidores e incluso de alguna ejecución o un error de seguridad se pueden alertar antes de tiempo de que se produzca.

Respecto a la aplicación web, existen varias técnicas, ya sea desde un hibernate o un spring (frameworks), ambos aplicando mvc, jsp, struts, javascript, html, bootstrap, css, etc., como se presenta en [5].

Existen varias técnicas para el almacenamiento y aquí se muestran dos muy efectivas [9] ya que actualmente son de las más usadas por su simplicidad y su bajo costo económico. También, actualmente se usan los servidores para guardar archivos, datos, etc., y para la trasferencia de archivos por su gran capacidad de procesamiento y por tener más espacio de almacenamiento.

Actualmente se trabaja con espacios inteligentes para la productividad digital y trabajo colaborativo, ya sea a través de sensores, programas elaborados con su respectiva base de datos, etc., como se muestra en el trabajo [2]

Justificación

La necesidad de almacenar grandes cantidades de datos relacionadas a un conjunto de usuarios y espacios resulta de suma importancia para el AISII de la UAM Azcapotzalco.

A fin de recolectar todos los datos relacionados a diversos espacios y usuarios, en este trabajo se propone diseñar un sistema de almacenamiento basado en archivos XML para su posterior explotación. El sistema está pensado para utilizar un medio de almacenamiento masivo externo y un protocolo de aplicación para la toma de decisión para la transmisión de información. La idea es guardar todos los datos y no carecer de esta capacidad de almacenamiento para no perder algún dato relevante.

Objetivos

Objetivo General

Diseñar e implementar un protocolo de aplicación enfocado en almacenamiento y transmisión de información en un sistema sensible al contexto, utilizando categorías de datos.

Objetivos Específicos

- Diseñar e implementar un módulo de división de datos por 2 categorías, primarios y secundarios, para su transferencia y almacenamiento.
- Diseñar e implementar un módulo de almacenamiento de datos en red utilizando un HDD, para poder enviar y ver la información desde cualquier computadora.

- Diseñar e implementar un módulo de adquisición de datos remoto en cada NC.
- Evaluar la integración de los módulos anteriores y medir el tiempo en que tarda en llenarse el HDD.
- Implementar y diseñar un módulo de control remoto de dispositivo central de la red, para la administración del mismo.

Marco Teórico

Sensores: Los sensores son aquellos dispositivos que nos ayudan a transformar una emisión del exterior, monitoreando o detectando variables y responder en consecuencia de aquellas magnitudes que puedan presentarse.

Servidor: Es una computadora o dispositivo capaz de recibir peticiones que realiza el cliente y de esta forma atenderlas a través del medio. Para esto se debe cumplir ciertas condiciones, ya sea la misma conexión, los protocolos, puertos, algunas credenciales, etc. También el servidor es capaz de almacenar archivos para posteriormente cuando se requieran se transfieran al cliente.

Multicliente: Se debe entender primero que un cliente no es más que, aquel que hace una petición a un servidor. Partiendo de esta pequeña definición se debe entender que un multicliente requiere de un centro para poder hacer a este mismo múltiples peticiones. Este centro es un servidor que va a responder estas peticiones para posteriormente responderlas.

Entonces se debe enfatizar que, un multicliente nos ayudar a trabajar en concurrencia para así poder agilizar la transferencia de datos.

Socket: Es una forma de comunicación entre cliente y servidor. Existen varias formas para lograr dicha comunicación, sin embargo, la más común es a través de un programa el cual cumple con la conexión de ambos lados. Los protocolos que usa son el TCP (Protocolo de control de transmisión) y UDP (Protocolo de datagramas de usuario) para la comunicación entre el cliente y el servidor, también se usa el HTTP (Protocolo de transferencia de hipertexto). Los sockets tienen un ciclo de vida, el servidor esperando a un cliente para establecer comunicación y un cliente buscando a un servidor para la misma comunicación.

Raspberry: Es un pequeño ordenador capaz de lograr múltiples tareas. Fue lanzado en el año 2012 y fue trabajado desde el año 2006 con el objetivo de fomentar la enseñanza de la computación a los niños. La Raspberry usa Arquitectura ARM, comúnmente se usa el sistema operativo Raspbian basado en Debian (GNU/Linux), sin embargo, hay muchos más sistemas que puede ser instalados en la placa. La Raspberry contiene puertos USB, HDMI, Memoria RAM, Ethernet, etc.

Base de Datos: Es aquel capaz de almacenar grandes cantidades de datos donde están agrupados o relacionadas de manera estructurada. La base de datos ayuda a los usuarios a registrar, eliminar o buscar la información que el mismo quiera sin redundancias e independiente de cualquier otra aplicación.

Python: Fué creado en el año 2006 por Guido van Rossum, es un lenguaje de programación que contiene multiparadigmas, desde una programación orientada a objetos hasta una programación reflexiva y también consigue el objetivo de ser multiplataforma. Su filosofía es tene run código visible y legible para cualquier usuario que quiera usarlo.

NAS: La NAS (Network Attached Storage), se puede considerar como una caja con varios discos duros, pero no solo eso, sino que también está conectado en la red. Su principal función es almacenar todos los datos de la red y también poder visualizar los datos, cambiarlos, etc. por personal autorizado o de confianza a través de credenciales.

XML: XML (Extensible Markup Language) es un lenguaje de etiquetas, además, es un formato universal adaptado al internet para datos estructurados. Se usa principalmente para la visualización y procesamiento de los mismos datos para diversas aplicaciones.

Control o Acceso Remoto: Nos ayuda para poder acceder a una computadora que se encuentra ubicada en algún otro lugar desde la red interna o externa, teniendo en cuenta que se pedirán permisos y credenciales para tener el control completo de la computadora o dispositivo que se quiera acceder.

MySQL: Es un lenguaje de consultas de datos que es usado también para gestionar una base de datos. MySQL tiene la ventaja de ser multiplataforma, esto quieres decir, que soporta varios sistemas operativos.

Desarrollo del Proyecto

Este proyecto tuvo que desarrollarse por 5 módulos.

La interacción de cada módulo se podrá ver reflejada en la Figura 1.

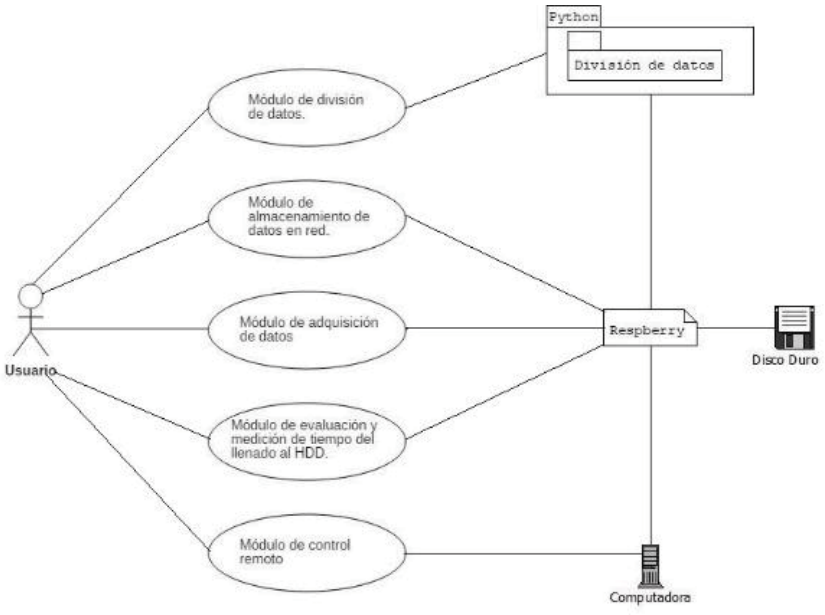


Figura 1: Diagrama de uso de los módulos

Módulo de división de datos

Se hizo un programa que, cada 5 segundos crea archivos XML, almacenándose en una Base de Datos. Estos archivos XML son divididos en dos carpetas, una carpeta se le puso nombre de "Importantes" y la otra carpeta recibió el nombre de "Secundarios". En la carpeta

"Importantes", se envían archivos que están fuera de rango a sus estándares, esto para que se tenga en cuenta que algo pudo haber pasado, y por ende se tiene que revisar y poner bastante atención a cualquier anomalía. En la carpeta de "Secundarios" se envían los archivos que están dentro de sus estándares, esto no quiere decir que no se tenga que revisar, por ejemplo, si hubo una abrupta diferencia en la temperatura de 5 °C ya sea que haya subido o bajado y sigue estando bajo el rango, se puede observar que hubo algún desequilibrio y se tiene que revisar.

Módulo de almacenamiento de datos en red (primer método)

En este módulo, se instaló el sistema operativo Raspbian con una memoria SD en la Raspberry. Posteriormente se viene una serie de pasos para que el sistema operativo arrancara desde un disco duro o incluso desde una USB, para que de esta forma se tenga más capacidad de almacenamiento dentro de una Raspberry usando un Disco Duro o una USB como se dijo anteriormente.

Los pasos a seguir fueron los siguientes:

Actualizamos el sistema e instalamos rpi (Raspberry Pi), en los sistemas más actuales ya está instalado

$ sudo apt-get update; sudo apt-get install rpi-update

$ sudo BRANCH=next rpi-update

Se habilita el USB boot mode, modificando la bandera OTP y reiniciamos

$ echo program_usb_boot_mode=1 | sudo tee -a /boot/config.txt

$ sudo reboot

Revisamos si se hicieron todos los pasos de manera correcta por lo que nos debería de aparecer 17:3020000a

$ vcgencmd otp_dump | grep 17:

17:3020000a

Modificamos de nuevo el fichero /boot/config.txt y comentamos la parte añadida, la cual es: program_usb_boot_mode=1, únicamente por si se quiere usar la SD en otra Raspberry.

$ sudo nano /boot/config.txt

Preparamos la USB particionandola en dos partes. La primera parte será con el formato fat32, donde será la parte booteable del sistema operativo y la segunda parte será
$ sudo parted /dev/sda

(parted) mktable msdos
Warning: The existing disk label on /dev/sda will be destroyed and all data on this disk will be lost. Do you want to continue?
Yes/No? Yes
(parted) mkpart primary fat32 0% 100M
(parted) mkpart primary ext4 100M 100%
(parted) print

Model: SanDisk Ultra (scsi)
Disk /dev/sda: 30.8GB
Sector size (logical/physical): 512B/512B
Partition Table: msdos
Disk Flags:

Number	Start	End	Size	Type	File system	Flags
1	1049kB	99.6MB	98.6MB	primary	fat32	lba
2	99.6MB	30.8GB	30.7GB	primary	ext4	lba

Se crea el boot y el root
sudo mkfs.vfat -n BOOT -F 32 /dev/sda1
sudo mkfs.ext4 /dev/sda2

Se copia el Sistema operativo (Raspbian) montando los ficheros target

sudo mkdir /mnt/target
sudo mount /dev/sda2 /mnt/target/
sudo mkdir /mnt/target/boot
sudo mount /dev/sda1 /mnt/target/boot/
sudo apt-get update; sudo apt-get install rsync
sudo rsync -ax --progress / /boot /mnt/target

Se regenera los host ssh

cd /mnt/target
sudo mount --bind /dev dev
sudo mount --bind /sys sys
sudo mount --bind /proc proc
sudo chroot /mnt/target
rm /etc/ssh/ssh_host*
dpkg-reconfigure openssh-server
exit
sudo umount dev
sudo umount sys
sudo umount proc

Se edita el fichero cmdline.txt para que arranca la USB como raíz en vez de la SD y también se edita fstab.

sudo sed -i "s,root=/dev/mmcblk0p2,root=/dev/sda2," /mnt/target/boot/cmdline.txt

sudo sed -i "s,/dev/mmcblk0p,/dev/sda," /mnt/target/etc/fstab

Se desmontan los targets y se apaga la Raspberry
cd ~

sudo umount /mnt/target/boot
sudo umount /mnt/target
sudo poweroff
Se desconecta la Fuente de alimentación, se saca la tarjeta SD y si todo salió bien y se siguieron los pasos arrancara con la USB.

Módulo de almacenamiento de datos en red (segundo método)

Los pasos a seguir son los mismos que en el primer método hasta el paso donde aparece la siguiente leyenda 17:3020000a

Actualizamos el sistema e instalamos rpi, en los sistemas más actuales ya está instalado

$ sudo apt-get update; sudo apt-get install rpi-update

$ sudo BRANCH=next rpi-update

Se habilita el USB boot mode, modificando la bandera OTP y reiniciamos

$ echo program_usb_boot_mode=1 | sudo tee -a /boot/config.txt

$ sudo reboot

Revisamos si se hicieron todos los pasos de manera correcta por lo que nos debería de aparecer 17:3020000a

$ vcgencmd otp_dump | grep 17:

17:3020000a

Modificamos de nuevo el fichero /boot/config.txt y comentamos la parte añadida, la cual es: program_usb_boot_mode=1, únicamente por si se quiere usar la SD en otra Raspberry.

$ sudo nano /boot/config.txt

Posteriormente, se instala el sistema operativo con la USB, proceso similar al realizado con la SD, se copian los archivos de NOOBS localizados en la página oficial de Raspbian y se arranca la Raspberry sin la SD y de esta forma arranca el sistema operativo en la Raspberry sin la tarjeta SD.

En la parte de la red se hizo una NAS, esto para que hubiera conectividad con toda la red y poder almacenar los archivos XML y no solo eso, sino que desde otra computadora se podría ver estos archivos que se almacenan en el sistema de almacenamiento que tuviera la Raspberry, además los archivos XML también se podrían enviar desde otra computadora y dejar la Raspberry.

Los pasos para lograr ese objetivo son los siguientes:

Como anteriormente se vio, sabemos en la USB está en /dev/sdax (donde x es un número), sin embargo, aquí cambia todo, pues está el sistema operativo en la USB o disco duro y

como sabemos tenemos el /dev/sda1 y el /dev/sda2, entonces se direccionaría en la ruta con la partición de ext4, ya que esta partición es la que más memoria tiene.

Creamos un directorio

sudo mkdir /media/USB

Editamos el archivo fstab y se pondrá la siguiente línea

sudo nano /etc/fstab

/dev/sda2 /media/USB vfat rw,user,auto,umask=000 0 0

Se instala Samba

sudo apt-get install samba samba-common-bin

Copiamos el archivo de configuracion ya que se va a modificar

sudo cp /etc/samba/smb.conf /etc/samba/smb.conf.old

Editamos y añadimos al final del archivo

sudo nano /etc/samba/smb.conf

[NombreNAS]
Comment=Disco USB
Path=/media/USB
Writeable=Yes
create mask=0777
browseable = Yes
valid users @users
force user=pi

Guardamos el archivo y además añadimos el usuario pi. Aparte se pone contraseña por seguridad y se reinicia samba.

sudo smbpasswd –a pi

sudo /etc/init.d/samba restart

Con esto ya tenemos una NAS dentro del disco duro o USB junto con el Raspbian funcionando.

Módulo de adquisición de datos remoto

En este módulo se obtiene un Servidor (nodo central) para que se conecte a varios clientes, donde estos clientes harán la petición y trabajaran de manera independiente. Un cliente será la temperatura, otro la humedad, etc. De esta forma se enviará la información al servidor y este los recibirá y los enviará a la base de datos.
Para lograrlo se tuvo que seguir la analogía y arquitectura de servidor multicliente, tomando en cuenta cosas fundamentales para que haya conexión, como los son el puerto, el host, el socket y la implementación de hilos para la concurrencia de cada archivo que se iba a estar enviando del cliente al servidor.

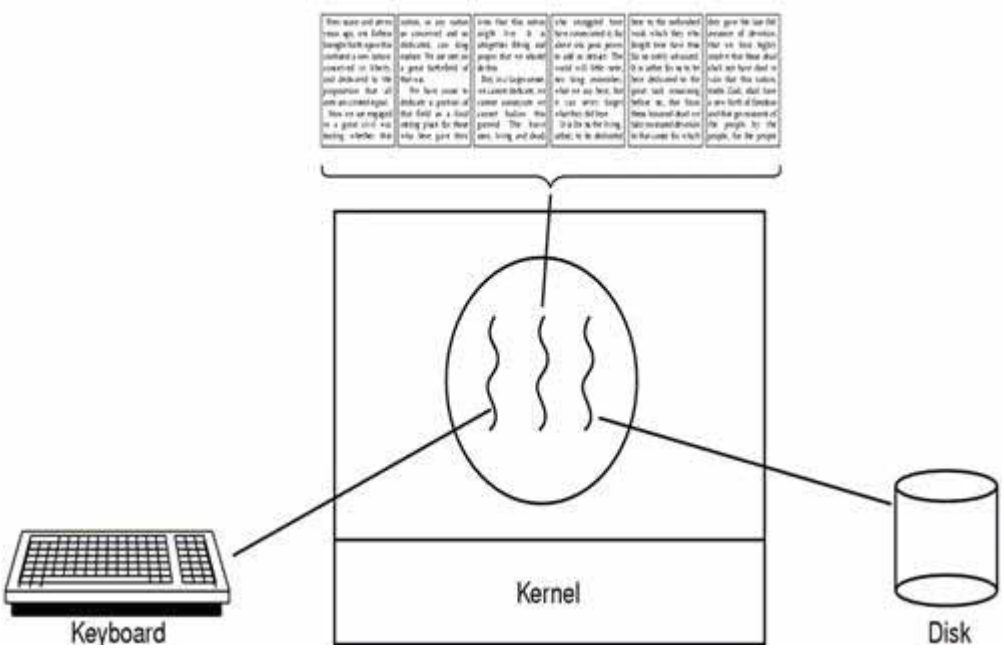


Figura 2: Multihilos trabajando concurrentemente [19]

Tomando en cuenta la Figura 2, se puede observar que se siguió prácticamente lo mismo, a partir de la computadora se creaban los archivos XML, los cuales en esta imagen están representados por la concurrencia que contienen información y posteriormente se almacenaban en el disco duro.

Evaluar la integración de los módulos anteriores

Aquí se hace una integración de todo lo trabajado y si está funcionando de manera correcta. Esto implica que el NAS funcione, el código funcione con su respectivo servidor y los muticlientes, que los archivos creados XML se almacenen en el Disco duro que está conectado en la Raspberry y que este mismo tenga el sistema operativo Raspbian y una Base de Datos

Módulo de control remoto de dispositivo central de la red

Para obtener control remoto en la Raspberry se usó el protocolo RDP, que únicamente consisten en dar los permisos a través de cualquier computadora que este en la misma red que la Raspberry.

Lo único que tenemos que hacer es instalar xrdp con el siguiente comando

sudo apt-get install xrdp

A través de una conexión remota de escritorio ponemos la IP de la Raspberry y nos abrirá una ventana en el que tendremos que dejar la opción Xorg que esta por defecto y únicamente tendremos que poner el usuario (pi) y la contraseña (raspberry) y con eso estaremos dentro de la Raspberry con otra computadora y podremos controlar la Raspberry por consecuencia de la conexión remota por si se requiere hacerle alguna configuración en algún futuro, sin tener que meternos directamente a la Raspberry y que siga trabajando con los datos

Resultados

Se dispondrá de una Raspberry Pi 3 Model B que es un ordenador pequeño de costo bajo y dispone de varios puertos fundamentales como USB 2.0, además, se le pueda dar varias funcionalidades según para lo que se quiera usar, desde un reproductor multimedia hasta un pequeño servidor.

Además, ya se garantiza la disponibilidad de los siguientes dispositivos, tanto Hardware como Software.

- Pantalla (HDMI) y su conector hacia la Raspberry
- Disco Duro con capacidad mínima de 1TB y puerto de comunicación USB 2.0
- Micro SD con capacidad de 8 GB
- Python 3.6.3 (Software libre)
- S.O. Raspbian Stretch 2.4.4 (Software libre)
- Samba 4.7 (Software libre)
- XRPD (Software libre)

Cuando se ejecuta el servidor para que se ponga en modo “escuchando”, se ejecuta también el cliente, en donde se puede observar como efectivamente se obtiene la arquitectura de servidor-multicliente; así mismo se puede ver como se conecta a una base de datos y además envía toda la información de los archivos que se van creando al servidor.

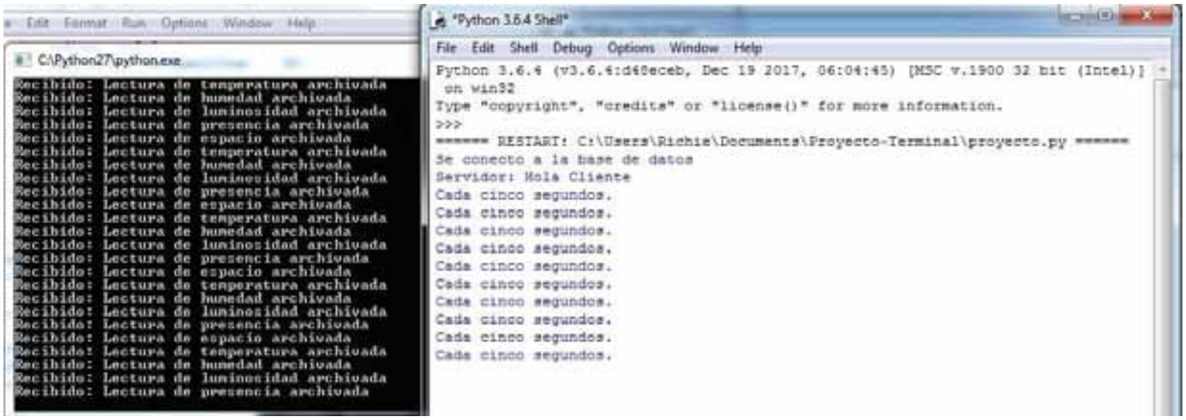


Figura 3: Ejecución de los programas cliente-servidor

En la Figura 3 se puede ver como se está enviando los archivos el cliente (cuadro blanco) cada 5 segundo y el servidor (cuadro negro) envía un mensaje de Recibido por cada archivo que recibe, ya sea de Temperatura, Humedad, etc.

Figura 4: Carpetas creadas y visualización de los programas

Cabe mencionar que los archivos XML creados están divididos en dos carpetas, en “Importantes” y “Secundarios” y estas carpetas también se les agrego otras carpetas para tener un mejor orden respecto a los XML creados, así como se puede visualizar en la Figura 4.

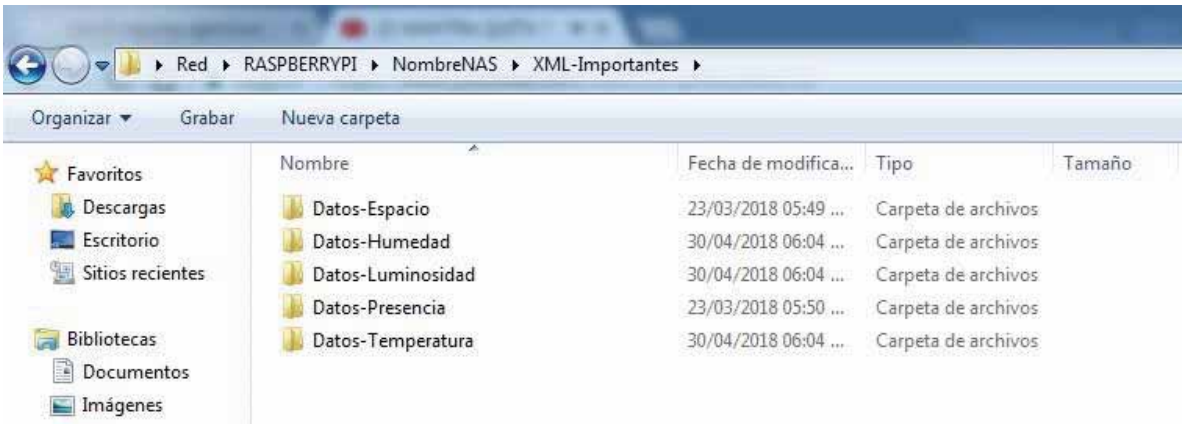


Figura 5: Carpetas de los datos que se irán almacenando los datos generados por la red.

Cada XML tiene su propio número que lo identifica y también sus respectivos datos, como se observa en la Figura 5. La división de datos se efectuó de acuerdo a estos mismos, determinando un rango "estándar" de acuerdo a los valores que se consideran normales.

```
<?xml version="1.0"?>
- <Datos_Temperatura>
  - <Temperatura_ID_2>
      <Lectura_de_Temperatura>18.0°C</Lectura_de_Temperatura>
      <Diferencia_con_la_Temperatura_Anterior>-31.04°C</Diferencia_con_la_Temperatura_Anterior>
      <Fecha_del_Registro>09/04/18</Fecha_del_Registro>
      <Hora_del_Registro>19:35:28</Hora_del_Registro>
    </Temperatura_ID_2>
</Datos_Temperatura>
```

Figura 6: Ejemplo del archivo XML creado.

En la Figura 6 se tiene el lenguaje de etiquetado XML, donde se puede ver el id, la temperatura, hora, fecha, etc.

Para la parte del Disco duro el cual iba a tener dentro el Sistema Operativo Raspbian los resultados no fueron los que se esperaban.

En el primer método mencionado dentro de ese módulo (módulo de almacenamiento de datos en red) el resultado se puede ver en la Figura 7:

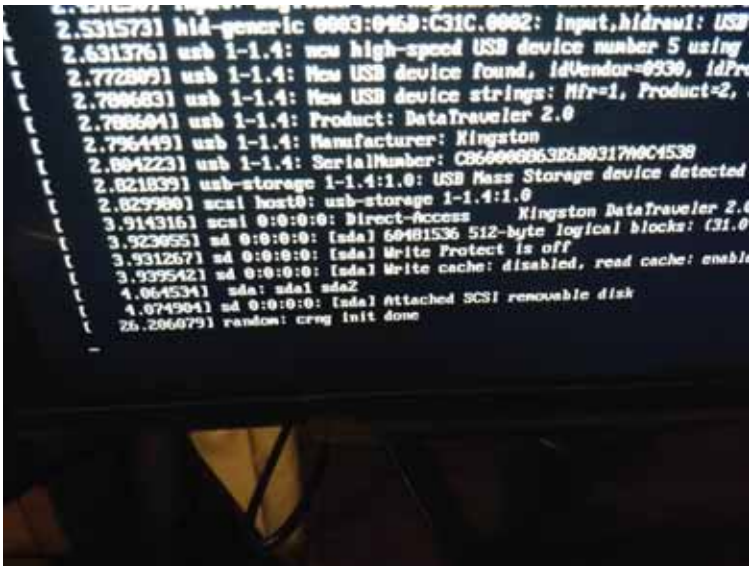


Figura 7: Falló de Raspbian en un disco duro

La leyenda “random: crng init done”, nos dice que puede haber tanto un error de Kernel, como también que le falta potencia a la Raspberry para que arranque desde un disco duro.

Para el segundo método de ese mismo módulo se tuvo lo siguiente:

Figura 8: Instalación del Raspbian sin SD y con la memoria USB

Como se puede observar en la Figura 8 si se logró instalar en una USB, un poco lento, pero con el sistema operativo dentro, sin embargo, en un disco duro no se puede, por más investigación elaborada no se sabe con certeza el por qué.

Sin embargo, esto no influye del todo en la elaboración de la NAS en una Raspberry. La instalación no lograda del Raspbian en un disco duro, no detiene la trasferencia de archivos, se puede usar, como se observó que se logró, en una USB en el la misma micro SD.

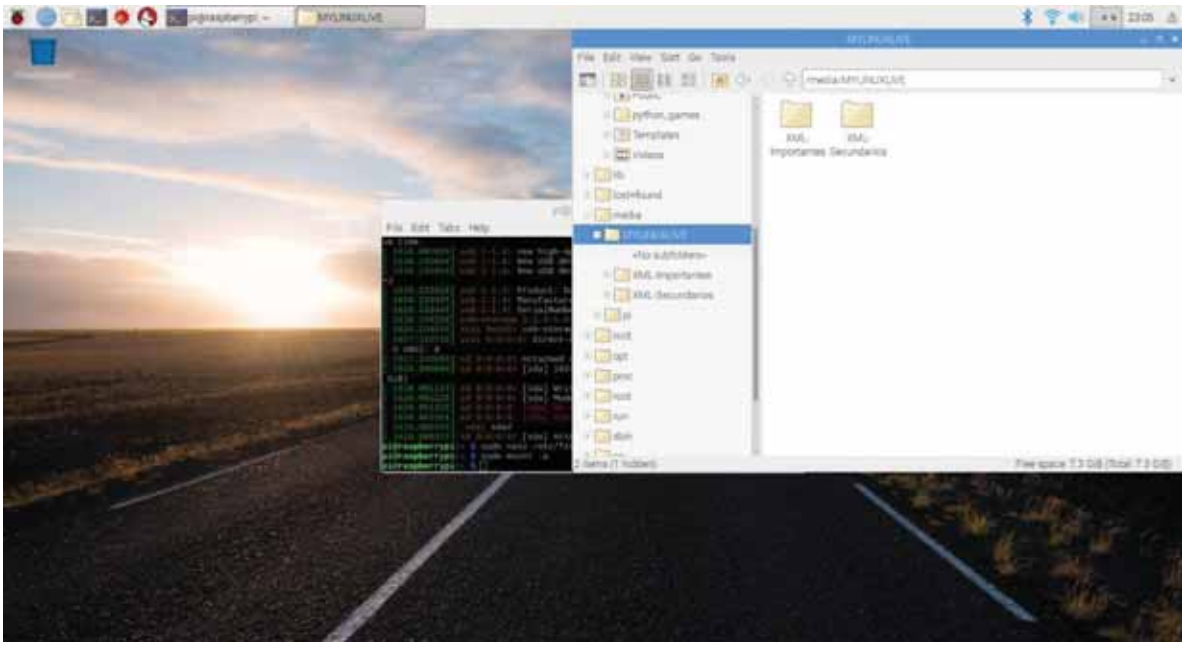


Figura 9: NAS

Los resultados que se obtuvieron fueron efectivos a comparación de los del módulo anterior. En la Figura 9, se observa como esta implementado el NAS y se pueden visualizar las carpetas que ya fueron creadas y los archivos se pueden visualizar.

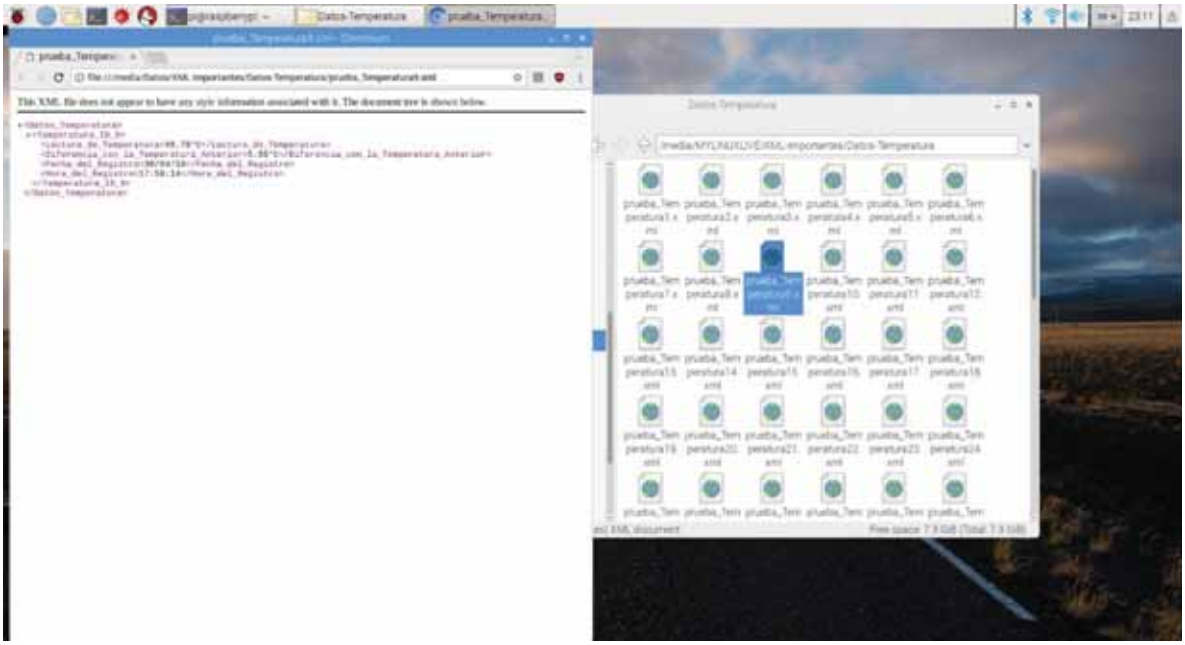


Figura 10: Archivos vistos en una USB creados en una computadora

Esto es lo que pasa desde la Raspberry, en la otra computadora lo que se puede visualizar es como está conectada en la red para la transferencia de archivos como se muestra en la Figura 10. Cabe mencionar que NombreNAS y MYLINUXLIVE es lo mismo solo que con diferente nombre, ambos direccionan hacia la USB, solo que NombreNAS, es el nombre que se le puso desde un inicio al archivo de configuración para que se montará la USB y transfiriera los archivos directamente y MYLINUXLIVE es el nombre de la USB.

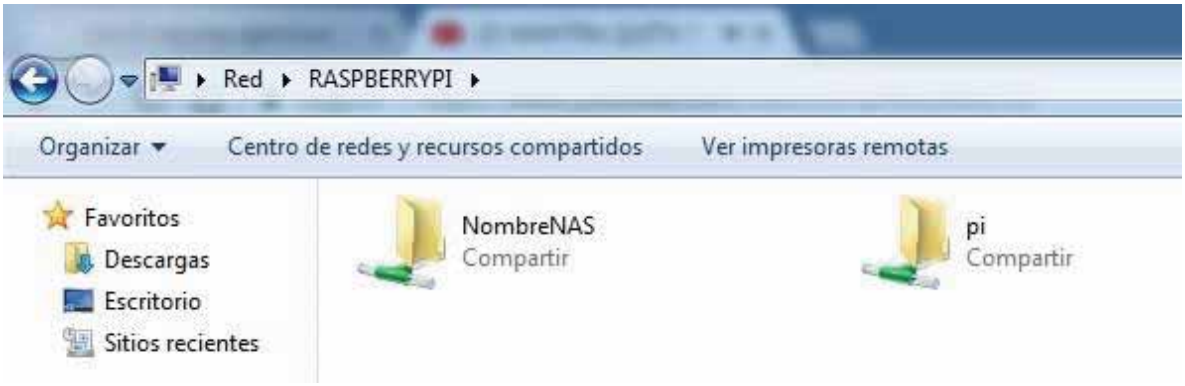


Figura 11: NAS desde un Windows.

Al igual que en la Raspberry, se pueden visualizar los mismos archivos, se pueden abrir y observar la misma información que en los XML, así como se ve en la Figura 11. Todos estos archivos como ya se sabe, son creados como se puede ver en el primer módulo y direccionándolo hacia la red para su respectiva creación de archivos, como se ve en la Figura 12.

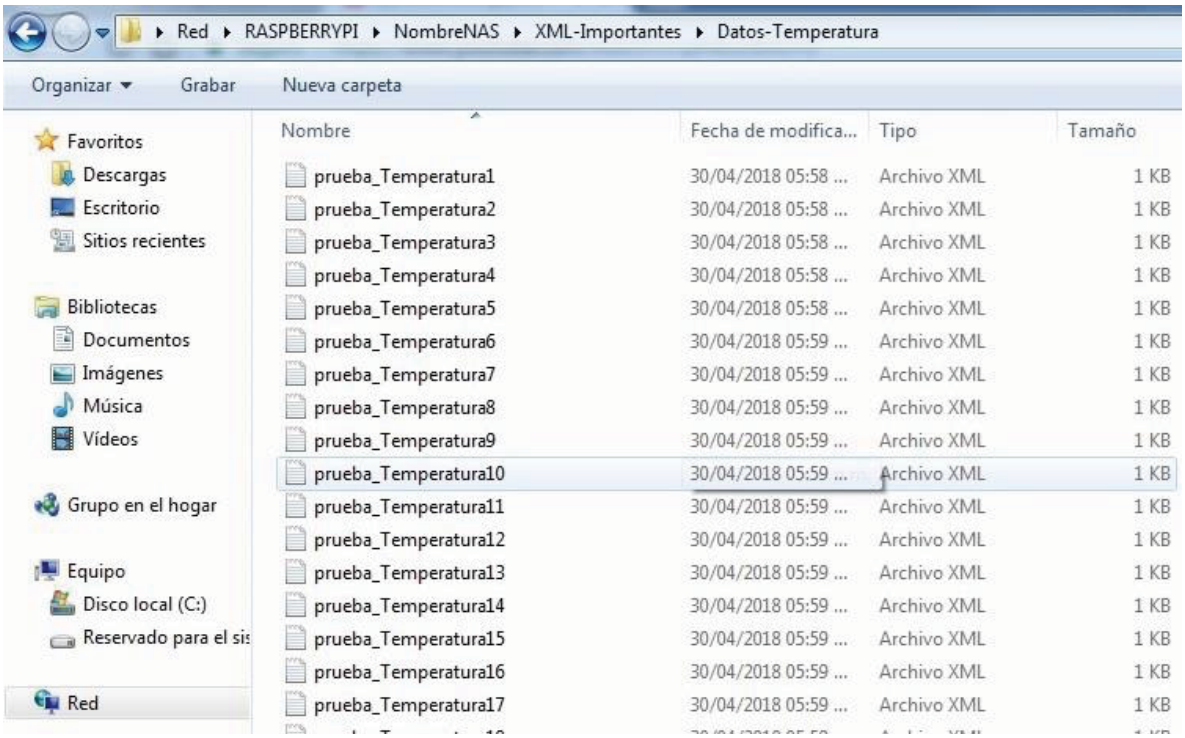


Figura 12: Archivos XML creados desde está misma computadora.

Y para finalizar, con XRPD se hace un control remoto desde la laptop para que así se deje implementado por completo la Raspberry y ya no se le tenga que mover a ninguna configuración directamente desde la Raspberry sino desde una computadora conectada a la misma red. Como se puede ver en la Figura 13, se tiene que poner una IP, esta IP es la de la Raspberry.

.

Figura 13: IP para la conexión remota

Posteriormente se siguen los pasos anteriormente mencionados y se obtiene control total de la Raspberry desde la misma computadora que se envían los archivos XML

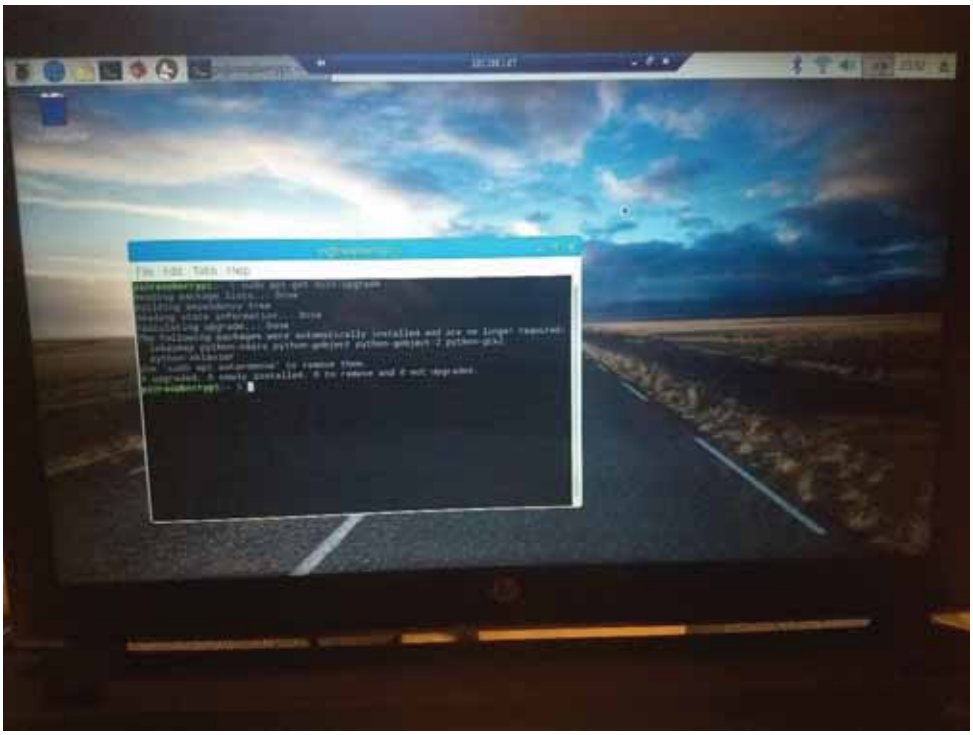


Figura 14: Control total de la Raspberry.

En la Figura 14 se puede ver como ya se tiene control total de la Raspberry desde una laptop. Se puede ver la diferencia porque en la parte de arriba se tiene la dirección IP de la Raspberry.

Análisis y Discusión de Resultados

Podemos observar que no se pudo obtener el sistema operativo Raspbian en un disco duro (ver Figura 7). El sistema operativo instalado se pudo lograr en una USB (ver Figura 8). Cabe mencionar que no en cualquier USB, sino en una en donde soporte el sistema operativo. Se desconoce las características específicas que debe tener dicha USB y también los discos duros, en dado caso que estos últimos si se logren instalar.

La implementación de esta técnica (instalar el sistema operativo Raspbian en un HDD) resulta no ser muy efectiva, aunque, sin embargo, se podía tener en el disco duro la base de datos, el sistema operativo, la NAS, los archivos XML y hacer la conexión remota hacia la Raspberry. Todo esto para utilizar el disco duro eficientemente y no olvidar que, además, cuenta con mucha más capacidad de almacenamiento.

Se debe seguir investigando y muy probablemente en un futuro no muy lejano se pueda implementar con más facilidad estas técnicas sin tantos errores o restricciones que tienen los dispositivos.

Sin embargo, todo lo demás se logró con éxito. Los archivos XML creados se verían mejor si solo estuvieran en un solo archivo en vez de crear tantos, se podría usar probablemente otros programas que faciliten estas implementaciones.

Conclusiones

En este proyecto se presentó una simulación de generación de archivos masivos, simulando un escenario de la vida real en un espacio inteligente. Se implementó una solución de almacenamiento basado en la red para el posterior análisis de datos. Para ello se implementó una red basada en arquitectura centralizada basada en el modelo cliente-servidor, teniendo varios clientes mandando información al servidor que trabajo como un nodo central. Además, al trabajar con un nodo central (servidor) y con varios clientes, congestiona la red, habrá saturación en la misma y por ende se necesitaría más recursos de procesamiento en el nodo central y de esta forma sería más práctico tener una arquitectura servidor-multicliente.

Por otra parte, se pudo también observar como la implementación del NAS es de gran utilidad ya que la información enviada se almacena en un dispositivo de almacenamiento y además se puede leer los archivos, y también ayuda a no saturar la red

Referencias

[1] E. I. Aguilar Suarez, "Explotación de base de datos geográfica para el proyecto de investigación Hábitat y Centralidad", proyecto de integración, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2015.

[2] R. Coulomb Bosc, M. T. Esquivel Hernández y G. Ponce Sernicharo, "Hábitat y centralidad en México un desafío sustentable", Centró de Estudios Sociales y de Opinión Publica, México, 2012.

[3] G. Aguilera Cortez, "Administración de información de un espacio inteligente", proyecto de integración, División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana Azcapotzalco, México, 2017.

[4] L. A. Lugo Hernández, "Sistema de almacenamiento masivo de datos mediante una memoria de alta densidad compact flash", División de Ciencias Básicas e Ingeniería, Universidad Autónoma Metropolitana, México, 2007

[5] J. A. Núñez Rodríguez, "Diseño e Integración de un Sistema de Adquisición de Datos Mediante el Uso de Arduino y Raspberry-Pi", Facultad de Ingeniería, Universidad Autónoma Nacional de México, México, 2014

[6] National Instruments Corporation, "National Instruments", http://www.ni.com/data-acquisition/what-is/esa/, 2017

[7] G. R. López Araiza "Sistema de análisis de grandes volúmenes de datos data bajo metodología ágil", Facultad de Ingeniería, Universidad Autónoma Nacional de México, México, 2017.

[8] An Enterprise Architect's Guide to Big Data, Oracle, 2016 http://www.oracle.com/technetwork/topics/entarch/articles/oea-big-data-guide-1522052.pdf

[9] J. P. Rodríguez Martínez, "Auditoria al proceso de almacenamiento de datos en la industria", Facultad de Ingeniería, Universidad Autónoma Nacional de México, México, 2010.

[10] IBM Knowledge Center, "IBM" https://www.ibm.com/support/knowledgecenter/es/ssw_ibm_i_72/rzam4/rzam4san.htm, 2017.
[11] Expert Designs "equisde" http://equisde.com/blog/raspberry-pi-3-arrancar-desde-usb/ , 2017
[12] Raspberry Pi Fundation "Raspberry" https://www.raspberrypi.org/documentation/hardware/raspberrypi/bootmodes/msd.md, 2017
[13] Raspberry Shop "Raspberry" https://www.raspberryshop.es/montar-un-nas-eficiente.php, 2017
[14] DeCodigo "DeCodigo" http://www.decodigo.com/python-crear-archivo-xml 2017
[15] Pledin "Plataforma Educatvia Informatica" https://www.josedomingo.org/pledin/2015/01/trabajar-con-ficheros-xml-desde-python_1/, 2015
[16] Pledin "Plataforma Educatvia Informatica" https://www.josedomingo.org/pledin/2015/01/trabajar-con-ficheros-xml-desde-python_2/, 2015
[17]StackOverFlow "StackOverFlow" https://es.stackoverflow.com/questions/62598/enviar-un-mensaje-al-cliente-desde-el-servidor-con-el-uso-de-sockets-en-python-3, 2018
[18] TutorialsPoint "SimpleEasyLearning" https://www.tutorialspoint.com/python/python_networking.htm, 2018
[19] Adrew S. Tanembaun, Sistemas operativos mdernos 3° edicion pag. 97 sección 2.2