Evaluación del modelo de referencia de "Internet of Things" (IoT), mediante la implantación de arquitecturas basadas en plataformas comerciales, open hardware y conectividad IPv6

Sandy E. Abasolo, Michelle A. Carrera, Rodolfo X. Gordillo, Carlos G. Romero

Abstract—En el presente proyecto se evalúa el modelo de referencia del Internet of Things, mediante el diseño e implementación de prototipos de red basados en las plataformas Arduino, Digi e IPv6. Cada prototipo emplea distintas tecnologías dentro de las capas del modelo. En los prototipos Arduino y Digi se realiza el monitoreo de parámetros del ambiente como temperatura y luminosidad y se ejecuta el control de encendido y apagado de dispositivos. Estas tareas se visualizan y gestionan desde una Interfaz web desarrollada y desde una nube de dispositivos propia, en el caso de Digi. El prototipo basado en IPv6 realiza el monitoreo del estado de dispositivos virtualizados IPv6, desde una interfaz, a la que se puede acceder dentro de una red nativa IPv6, que para este proyecto es la red de CEDIA, o mediante un proveedor de túneles IPv6 sobre IPv4. Por ´ultimo se analiza que los prototipos se acoplen al modelo de referencia y se establecen las diferencias entre ellos.

Index Terms — Internet of things, Arduino, Digi, IPv6, Android, CEDIA, Monitoreo, Gestión.

I. INTRODUCCIÓN

Hasta el momento la atención se ha centrado en la interacción humana a través de las tecnologías de información y comunicación, pero en la actualidad la comunicación entre dispositivos se encuentra en constante aumento y se han desarrollado nuevos conceptos como el Internet of things, que representa la evolución del Internet, permite relacionar objetos con objetos y objetos con personas, los cuales se pueden conectar, comunicar y controlar a través de la red mediante una nube de servicios (Cloud service), de tal forma que la información que se obtiene se encuentre disponible en cualquier lugar y en todo momento.

Una de las principales tecnologías empleadas en los trabajos implementados sobre IoT son las Redes WSAN, las cuales facilitan su desarrollo, dentro de las principales se encuentra la red Zigbee que ofrece estándares inalámbricos y globales, para conectar una amplia gama de dispositivos.

Una nueva tendencia para realizar la gestión de dispositivos dentro de IoT, son elementos direccionables, que adoptan el protocolo IP, y permiten la interacción de miles de dispositivos, que no requieren de intermediarios para acceder a la red y con los que se puede lograr una hiperconectividad.

En este proyecto se evaluará el modelo de referencia de "Internet of things" (IoT) mediante la implantación de arquitecturas basadas en plataformas comerciales (Digi), Open Hardware (Arduino) y conectividad IPv6, para la gestión remota de dispositivos en la nube.

II. MARCO TEÓRICO

A. *Concepto de Internet of Things*

El centro de investigación SAP define el Internet of things como" Un mundo en el que los objetos físicos están perfectamente integrados en la red de información, y donde los objetos físicos pueden llegar a ser participantes activos en los procesos de negocio. Los servicios están disponibles para interactuar con

S.E. Abasolo, es graduada de Ingeniera en Electrónica y Telecomunicaciones en la Universidad de las Fuerzas Armadas Espe,Sangolquí-Ecuador, (email:sandyestefany89@gmail.com).
M.A. Carrera, es graduada de Ingeniera en Electrónica y Telecomunicaciones en la Universidad de las Fuerzas Armadas Espe, Sangolquí-Ecuador, (email:michelle4651@gmail.com).
R.X Gordillo, es profesor del Departamento de Eléctrica y Electrónica en de en la Universidad de las Fuerzas Armadas Espe, Sangolquí- Ecuador, (email:rxgordillo@espe.edu.ec).
C.G. Romero, es profesor del Departamento de Eléctrica y Electrónica en de en la Universidad de las Fuerzas Armadas Espe, Sangolquí- Ecuador, (email:cgromero@espe.edu.ec).

estos" objetos inteligentes" a través de Internet, consultar y modificar su estado, y toda la información asociada a ellos, teniendo en cuenta la seguridad y la privacidad." [1].

B. Modelo de Referencia

El modelo de referencia del IoT está compuesto de cuatro capas: [2]

- Capa de Aplicación.
- Capa de Servicio.
- Capa de Red.
- Capa de Dispositivo.

Capa de Aplicaciones: Pone en uso la gran cantidad de información creada a partir del IoT y es donde se encuentra el mayor potencial.

Capa de Servicio: Definida como computación en la nube, costa de dos grupos:

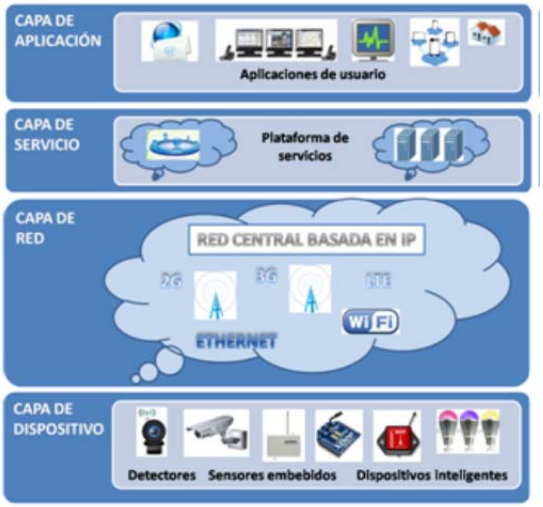

Figura 1: Modelo de referencia del IoT.

Funciones de soporte general: Son funciones que pueden ser utilizadas por diferentes aplicaciones, como procesamiento o almacenamiento de datos.

Funciones de soporte específico: son funciones especiales que se adaptan a los requisitos de las aplicaciones, con el fin de proporcionar diferentes funciones de apoyo a las diferentes aplicaciones del IoT.

Capa de Red: Es la infraestructura de redes alámbricas e inalámbricas, consta de los dos siguientes tipos de funciones:

- **Función de Red:** proporciona funciones de control, como el acceso y el control de recursos de transporte, la gestión de la movilidad, la autenticación, autorización y contabilidad (AAA).
- **Función de Transporte:** es responsable de proporcionar conectividad para el transporte del servicio del IoT y la información de datos específicos de la aplicación, así como el transporte de la gestión de control.

Capa de Dispositivo: La capa de dispositivo conecta todo con el Internet y es la infraestructura clave para el IoT. Consiste en la tecnología inalámbrica y la interfaz del sensor para recolectar y transmitir señales análogas/digitales al controlador principal.

La tecnología inalámbrica está basada en IEEE 802.15.4. Esta capa realiza dos funciones:

- Función de dispositivo.
- Función de Gateway.

Las cuatro capas se encuentra en la capacidad de trabajar en función de administrar y dar seguridad, las cuales en conjunto forman el modelo de referencia del Internet of Things.

III. MATERIALES Y M´ETODOS

A. Requerimientos de construcción del IoT para redes tradicionales

- Detección y Recolección de datos.
- Capacidad de procesamiento embebido.
- Capacidad de comunicación alámbrica y/o inalámbrica.
- Software para automatizar.
- Procesamiento remoto y acceso a la nube.

B. Arquitectura de las redes tradicionales

La aplicación del modelo de referencia IoT mediante las plataformas Arduino y Digi consiste en la monitorización de ambientes y el control de dispositivos, para lo cual se cuenta con cuatro nodos, los nodos de monitoreo recolectan datos de sensores de temperatura, humedad y luminosidad en cada ambiente. Y los nodos de control permiten el encendido y apagado de dispositivos. La

comunicación entre los nodos es mediante Zigbee, uno de los nodos realiza las funciones de coordinador.

El coordinador recolecta los datos de los nodos, realiza un procesamiento local y posteriormente transmite esta información hacia la interfaz web, para su visualización y procesamiento.

La interfaz web utiliza el servicio de alojamiento web, Google App Engine, este servicio permite ejecutar aplicaciones sobre la infraestructura de Google [3]. En la interfaz del prototipo basado en la plataforma Arduino, se visualizan los datos de los sensores, el tiempo de la muestra, el valor medio y la varianza, dentro de una tabla y en un gráfico que plasma estos valores. También se puede realizar el encendido o apagado de dispositivos en otra interfaz.

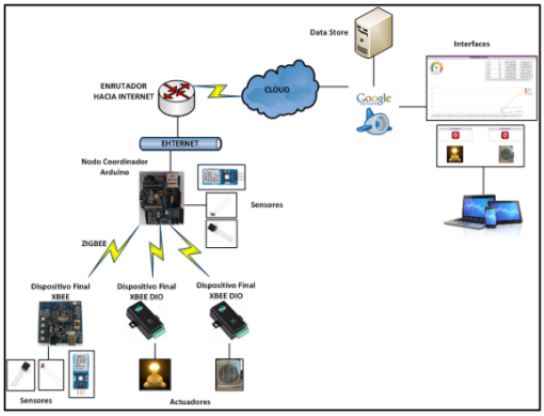

Figura 2: Arquitectura del prototipo Arduino.

En el prototipo basado en la plataforma Digi, la visualización de los datos de los sensores se puede realizar tanto en la interfaz de Etherios Device Cloud propia de Digi, que adicionalmente realiza el procesamiento de los mismos, como en una aplicación de cliente alojada en el servicio Google App Engine. En la interfaz se visualizan los datos de los sensores, y el tiempo la muestra en un gráfico. También se puede realizar el encendido o apagado de dispositivos.

C. Diseño e implementación de los prototipos de red tradicionales

La red de sensores Zigbee en estos prototipos está compuesta por cuatro nodos, un Coordinador y tres nodos terminales, uno de monitoreo y dos de control. La topología de la red implementada es tipo estrella.

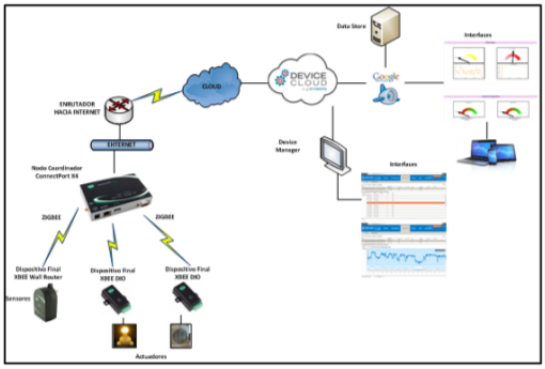

Figura 3: Arquitectura del prototipo Digi.

El coordinador y los dispositivos finales seleccionados corresponden a módulos XBEE Series 2. El dispositivo final de monitorización en el prototipo Arduino utiliza dos entradas analógicas para la lectura de sensores. En el caso de Digi los sensores se encuentran embebidos en el dispositivo Wall Router. Los dispositivos finales de control utilizan una salida digital para el control de encendido y apagado.

El coordinador en el prototipo Arduino, está compuesto por una tarjeta Arduino UNO, una Arduino EthernetShield, que permite la conexión vía Ethernet a un enrutador, que le proporcione acceso a Internet. Para conectarse de forma inalámbrica con los dispositivos finales, requiere de una XBEE Shield, que contendrá al módulo XBEE. El coordinador utilizará una entrada digital y una analógica para la lectura de sensores.

El coordinador en el prototipo Digi corresponde a un dispositivo ConnectPort X4 que no requiere ningún montaje, realiza la conversión de red Zigbee a red Ethernet para acceder a Internet.

Los sensores utilizados para medir la temperatura, humedad y luminosidad en el caso del prototipo Arduino son:

- Fotocelda LDR
- LM35
- DHT11

El sensor LDR requiere de un circuito de acondicionamiento para la adquisición de los datos

que provee. El resto de sensores no necesitan un circuito o ya lo tienen incorporado.

Los dispositivos finales de control corresponden a equipos XBEE DIO, para los dos prototipos, requieren circuitos de acondicionamiento para la activación de actuadores que realicen el encendido y apagado de dispositivos.

El programa desarrollado en el coordinador (Arduino), lee los datos de las entradas analógicas y digitales, tanto del coordinador como del dispositivo final de monitorización, realiza un pre-procesamiento de los valores obtenidos, conecta al coordinador con las aplicaciones Google App Engine y viceversa enviando y recibiendo datos, permitiendo el control de dispositivos.

El coordinador en el prototipo Digi utiliza el software DIA para la conexión de los dispositivos finales e integración a la red.

D. Interfaz de usuario de los prototipos de red tradicionales

El desarrollo de las interfaces en este proyecto se realiza en el lenguaje de programación Python y se encuentran alojadas en la plataforma Google App Engine.

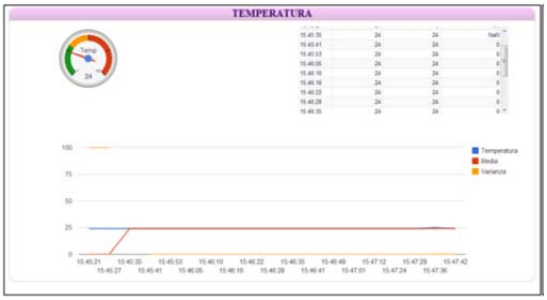

Figura 4: Interfaz de monitoreo en el prototipo Arduino.

En la interfaz del prototipo Arduino se pueden visualizar tres pestañas, las dos primeras corresponden al monitoreo de dos ambientes y la tercera al control de dispositivos.

En la ventana de control se pueden visualizar dos botones que al presionarlos controlan el encendido y apagado de dispositivos.

Figura 5: Interfaz de control en el prototipo Arduino.

Los datos generados por los dispositivos conectados al coordinador en el prototipo Digi, se pueden visualizar en la pestaña Data Streams de Etherios Device Cloud. El Device Cloud realiza un procesamiento de los datos obtenidos y los presenta en gráficas.

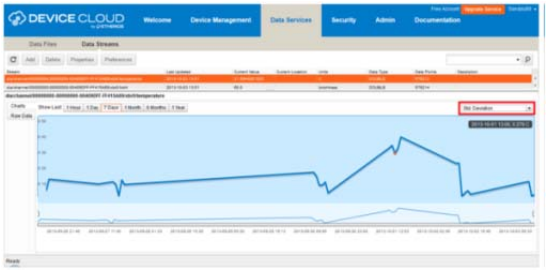

Figura 6: Interfaz de Etherios Device Cloud

La aplicación de cliente de este prototipo, contiene dos secciones, una para el monitoreo y una segunda que controla dos dispositivos.

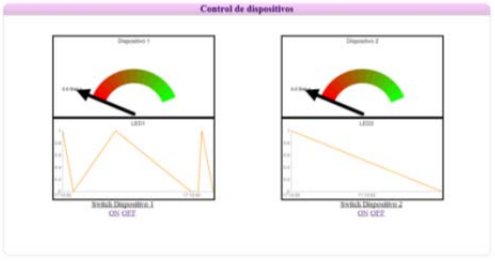

Figura 7: Interfaz de control en el prototipo Digi

Las aplicaciones fueron diseñadas dentro de la arquitectura Cliente - Servidor REST (Representational State Transfer), en donde el cliente inicia peticiones hacia el servidor, el servidor las procesa y retorna respuestas apropiadas.[4]

En la aplicación del prototipo Arduino, se realizan operaciones GET periódicamente para muestrear el estado de un recurso.

En la aplicación del prototipo Digi el cliente inicia peticiones POST hacia el servidor. La aplicación se comunica con Etherios Device Cloud mediante el método RCI Request. [5]

En la ventana de control se pueden visualizar dos botones que al presionarlos controlan el encendido y apagado de dispositivos.

E. Requerimientos de construcción del IoT con IPv6

- Sistemas inteligentes de detección y recolección de datos.
- Conectividad extremo a extremo.
- Compatibilidad de direccionamiento.
- Compatibilidad de internet.
- El sistema debe proporcionar una API.[6]

F. Arquitectura de la red IPv6

La arquitectura basada en dispositivos IPv6, consiste en la monitorización del estado de dispositivos. Se dispone de tres nodos, que se representan mediante máquinas virtuales con el sistema operativo TinyCore, que posee soporte IPv6 e incluye un servidor SSH [7], debido a escases de sensores IPv6, de los cuales se obtienen datos de estado de memoria, estado de disco, procesos ejecutados, estado de puertos y se realiza un control de estos dispositivos cerrando procesos activos. Los nodos se conectan a una red IPv6 a través de Red CLARA, para este proyecto se utilizó la red CEDIA.

El control de los dispositivos y la visualización de los datos se realiza de dos formas: la primera consiste en una interfaz web, que concentra la información de todos los dispositivos finales. En la segunda forma la interfaz web se encuentra en servidores Apache en cada uno de los dispositivos y se accede a ella mediante la dirección IPv6 de cada nodo. En la interfaz se pueden visualizar los datos de los parámetros ya mencionados de los dispositivos IPv6 y un gráfico que plasma estos valores. También se puede realizar el control de los procesos que se encuentran ejecutando los dispositivos.

Se puede acceder al servicio web mediante dos formas:

- El usuario puede acceder dentro de una red nativa IPv6.

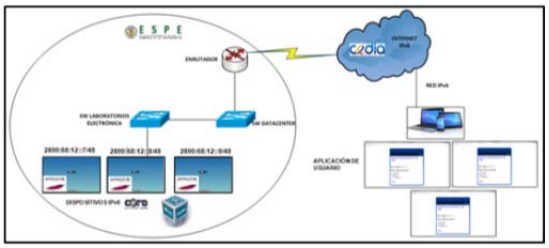

Figura 8: Arquitectura del prototipo IPv6 con red nativa.

- El usuario puede acceder mediante un proveedor de túneles de red IPv6 sobre IPv4.

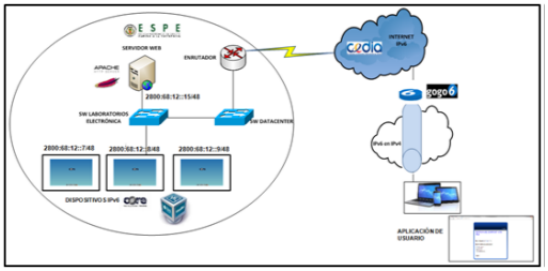

Figura 9: Arquitectura del prototipo IPv6 con acceso por túnel.

G. Interfaz de usuario del prototipo IPv6

En esta plataforma se monitorea el estado de dispositivos IPv6 (máquinas virtuales), por lo cual, se desarrolló una interfaz que obtiene datos de: Espacio de memoria, Procesos activos, Espacio de disco, Estado de puertos y se controla el cierre de procesos. La recolección de datos se realiza mediante la ejecución de comandos de consola a la cual se accede mediante SSH.

En la página principal de la interfaz del servidor externo se puede elegir el dispositivo y el proceso a realizar. En la página principal de la interfaz de los servidores de cada nodo se indica el nodo al que se ha accedido y las opciones a realizar. Para las opciones de Estado de memoria y Espacio de disco,

adicionalmente, en una tercera ventana, se puede visualizar un gráfico de tipo pie.

La interfaz de usuario desarrollada se aloja en un servidor HTTP apache y se la realizó en el lenguaje de programación PHP, que utiliza la librería libssh2 para el acceso remoto a dispositivos mediante el protocolo SSH2.

1) **Acceso al servidor mediante un túnel IPv6:** En este proyecto se utiliza la aplicación gogoCLIENT como proveedor de túnel IPv6 sobre IPv4. Establece un túnel con la red gogo6 proporcionándole al usuario una dirección IPv6 Figura 10: Página principal de la interfaz del prototipo IPv6 alcanzable desde Internet. GogoCLIENT establece y mantiene el túnel mediante el protocolo propietario TSP. [8]

El modo de encapsulación de túnel utilizado es v6udpv4 que realiza un encapsulado de IPv6-in-UDP-in-IPv4, diseñado para trabajar a través de NAT.

Figura 10: Página principal de la interfaz del prototipo IPv6.

Figura 11: Resultado de la ejecución del comando SSH.

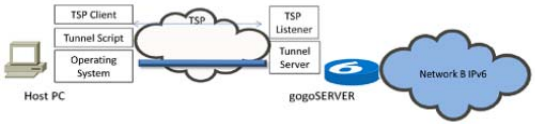

Figura 12: Funcionamiento del túnel IPv6 sobre IPv4.

IV. PRUEBAS Y EVALUACI´ON

A. Evaluación del modelo de referencia IoT en el prototipo Arduino

- **Capa de Aplicación y Servicio:** La capa de servicio ofrece un software como servicio basado en la arquitectura Cloud Computing. La aplicación se realizó en el dominio de Google App Engine, en donde se puede visualizar y acceder a los datos ubicuamente. La aplicación permite realizar el procesamiento de los datos obtenidos.

- **Capa de Red:** El coordinador de la red recibe los datos, que provienen desde los dispositivos de forma inalámbrica y envía esta información hacia la nube de Google en una red IPv4. El protocolo de transporte utilizado es TCP.

- **Capa de Dispositivo:** En la capa de dispositivo se encuentran los sensores que interactúan con su coordinador en la red Zigbee, para la transmisión y recepción de datos.

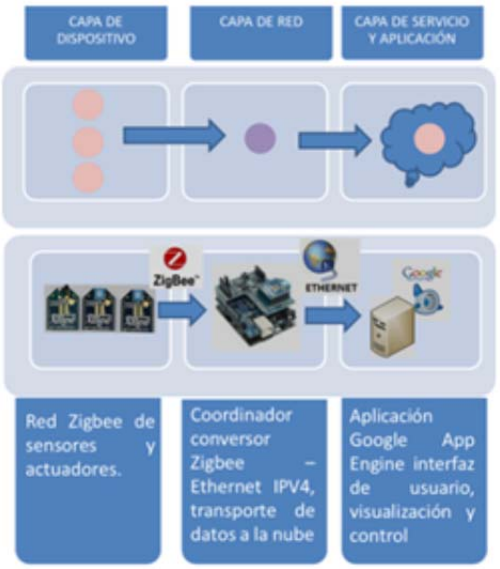

Figura 13: Modelo de referencia IoT aplicado al prototipo Arduino.

B. Evaluación del modelo de referencia IoT en el prototipo Digi

- **Capa de Aplicación:** La aplicación se realizó en el dominio de Google App Engine en la cual se puede visualizar y acceder a los datos obtenidos desde la nube Etherios, mediante XML y HTTP.

- **Capa de Servicio:** En la capa de servicio la plataforma comercial Digi ofrece una plataforma como servicio llamada Etherios, en el cual se puede visualizar todos los dispositivos conectados a la red zigbee y el procesamiento de los datos provenientes de los mismos.

- **Capa de Red:** El software que permite a los sensores interactuar con su coordinador Connect Port X4 en la red Zigbee, se llama Device Integration Application (DIA). La infraestructura de red utilizada es alámbrica e inalámbrica. El protocolo de transporte es TCP.

- **Capa de Dispositivo:** En la capa de dispositivo se encuentran sensores embebidos de la marca Digi, sirven para la detección y recolección de datos. La tecnología inalámbrica utilizada en esta capa es Zigbee.

C. Evaluación del modelo de referencia IoT en el prototipo IPv6

- **Capa de Aplicación y Servicio:** La aplicación se realizó, en un servidor web Apache que permite visualizar y acceder a los datos ubicuamente desde cualquier dispositivo conectado a internet IPv6, mediante SSH. Visualizando el estado de los dispositivos y controlando sus procesos. La aplicación se realizó con el esquema tradicional de servicios REST sobre HTTP/TCP.

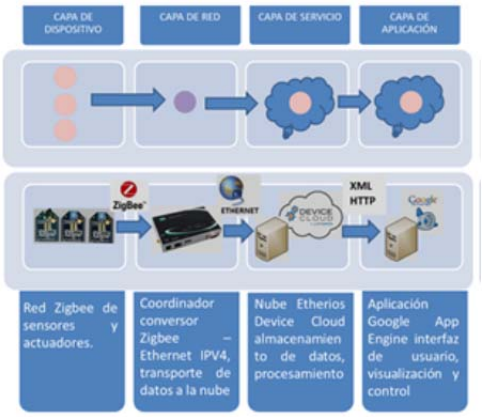

Figura 14: Modelo de referencia IoT aplicado al prototipo Digi.

- **Capa de Red y Dispositivo:** En la capa de dispositivo se tienen máquinas virtuales con sistema operativo Linux que representan a sensores con soporte IPv6, es el protocolo de red utilizado en esta plataforma, por lo tanto la capa de dispositivo corresponde a una red Ethernet y la capa de red es IPv6. Estos dispositivos se conectan directamente con internet IPv6 sin la necesidad de pasarelas propietarias ni concentradores y ofrecen servicios básicos como el servicio web.

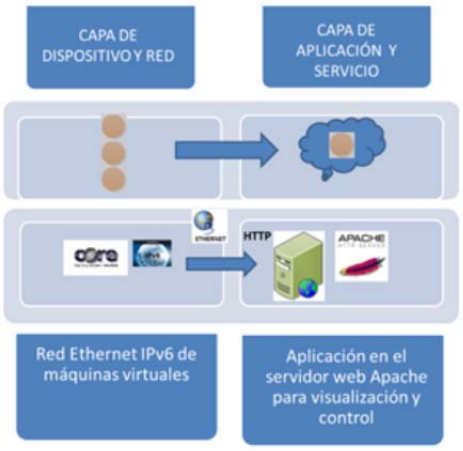

Figura 15: Modelo de referencia IoT aplicado al prototipo IPv6.

D. Análisis de alcance en la red Zigbee de los prototipos

Se realizaron pruebas mediante el software X-CTU, midiendo la potencia de la señal recibida y el rango del alcance de la señal, verificando cuantos paquetes llegan buenos y cuantos no. El parámetro medido es RSSI, que indica el nivel de potencia de la señal recibida en redes inalámbricas. Su rango se encuentra entre:

- 0 dBm - Señal ideal.
- -90 dBm a -100 dBm Enlace malo.

Como podemos observar, a mayor distancia entre nodos, la potencia recibida en los módulos XBee es menor e incrementan el número de paquetes perdidos. De acuerdo a las especificaciones técnicas de los módulos, tienen un alcance en interiores de 40m. [9] En la práctica de este proyecto se determina que el alcance es mucho menor en interiores y sin línea de vista. La distancia máxima alcanzada, aproximadamente es de 12m.

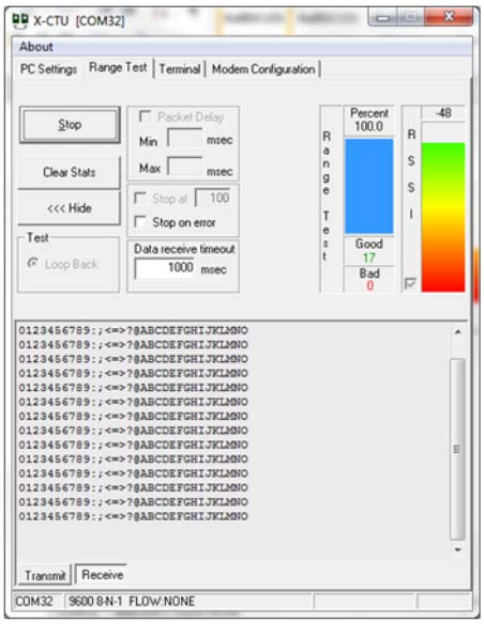

Figura 16: Medición de Range Test en el software X-CTU.

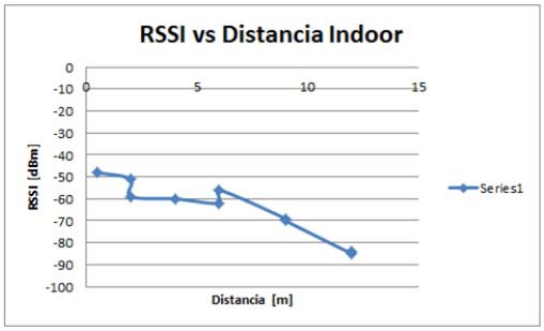

Figura 17: Nivel de potencia de la señal recibida vs. la distancia.

E. Análisis comparativo del tiempo de encendido y apagado de dispositivos

Se realizaron pruebas con el prototipo realizado Arduino, con el prototipo Digi y con una plataforma comercial que provee dispositivos embebidos de la marca Belkin, que permiten realizar el control de encendido y apagado de dispositivos en un ambiente wireless, trabaja con cualquier red wi-fi y no requiere de ninguna programación ya que provee una aplicación Android llamada WeMo. [10]

Figura 18: Interruptores Wifi de la marca Belkin.

Las pruebas se realizaron en una red con velocidad de descarga de 2.17 Mbps y velocidad de carga de 0.51 Mbps.

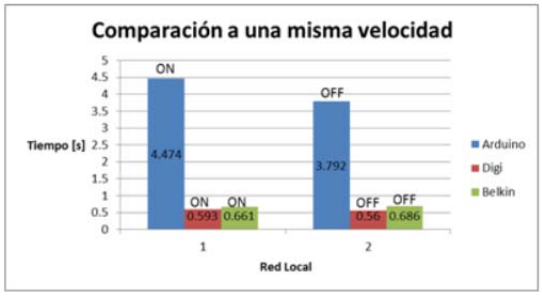

Figura 19: Comparación del tiempo de respuesta entre los prototipos Arduino, Digi y los dispositivos Belkin.

Se puede observar que el prototipo Arduino presenta el mayor retardo para el control de dispositivos, debido a que la interacción con los nodos de control es mediante lectura de tramas, su arquitectura fue ensamblada manualmente y la programación es más compleja que los dispositivos comerciales. La eficiencia depende de la velocidad de transmisión de la red para todos los prototipos.

F. Comparación de las Características del Modelo de Referencia entre los prototipos

- En los prototipos IoT tradicionales no se puede acceder directamente a los dispositivos finales, sino solo a través de su coordinador, mientras que en el prototipo basado en ipv6 el acceso al dispositivo es directo y mediante su dirección IP, que es su identificador dentro de la red y es ´único, permitiendo mayor facilidad de administración.

- Los dispositivos en el prototipo IPv6 vienen integrados con todas las capas del modelo IoT, incluyendo sensores embebidos, modulo de comunicación, y servicios de usuario. Es decir que son dispositivos inteligentes con mayores capacidades, facilitando su implementación.

- En el prototipo IPv6 no se requiere de un dispositivo intermedio (Gateway) que concentre la información de los dispositivos finales, debido a que los sensores se encuentran conectados directamente a la red de Internet IPv6.

- En el prototipo basado en Digi, al poseer una nube de dispositivos propia, llamada Etherios, reduce la complejidad de la obtención y visualización de los datos de la red de dispositivos y permite una mejor administración de los nodos. El retardo en el acceso a los datos es menor que en una nube pública como la que se utiliza en el prototipo basado en la plataforma Arduino.

- El prototipo basado en IPv6 permite una conectividad end-to-end, a diferencia de los prototipos tradicionales (Digi, Arduino) que utilizan el protocolo IPv4, ya que elimina la barrera de NAT, con las direcciones IP específicas y permanentes en los dispositivos. Añadiendo así la capacidad de movilidad, permitiendo cambiar de puntos de acceso a internet sin perder conexión. [11]

- Una de la más importantes diferencias entre las redes tradicionales e IPv6 es la gran dimensión de las tablas de encaminamiento en la red troncal de Internet que utiliza IPv4, que la hace ineficaz y perjudica considerablemente los tiempos de respuesta. En IPv6 el encaminamiento en la red troncal es más eficiente, debido a una jerarquía de direccionamiento basada en la agregación y a que la fragmentación y desfragmentación de los paquetes se realiza extremo a extremo

y no en los nodos intermedios la cual mejora su tiempo de respuesta. [12]

- En el acceso a la interfaz web mediante un túnel IPv6 sobre IPv4, el proceso de transmisión de los paquetes inicia en un extremo del túnel, luego son encapsulados y enviados hasta el enrutador del proveedor de túnel, el cual lo desencapsula y por ´ultimo se inicia el enrutamiento hacia el destino final en la nube de IPv6. Este proceso incrementa el retardo en un 50% de una red nativa. [13] Por lo tanto una red IPv6 nativa mejora la velocidad de transmisión.

- A diferencia de las redes tradicionales, el prototipo IPv6 permite una mayor escalabilidad horizontal, debido a que es posible añadir gran cantidad de dispositivos a la red, ya que IPv6 tiene millones de direcciones disponibles para los dispositivos. Por lo tanto en las redes tradicionales se dificulta el desarrollo de una red IoT.

- La implementación mediante la plataforma Arduino al ser Open Hardware es más económica, pero su construcción es más compleja al ser manual y carece de soporte técnico de fabricante, a diferencia de la plataforma comercial Digi. En el prototipo basado en IPv6 no es posible estimar costos debido a la falta de disponibilidad de dispositivos IPv6.

Los aspectos mencionados tienen una valoración de 5 para el prototipo que mejor cumpla con cada parámetro y de 0 como mínima valoración y se presentan en la Tabla1:

Tabla I: Valoración de los prototipos.

Parámetro de Evaluación	Prototipo Arduino	Prototipo DIGI	Prototipo IPv6
Disponibilidad de información	5	3	1.5
Disponibilidad de equipos	5	5	1
Costos de Implementación	4	2.5	2
Facilidad de Implementación	2	5	5
Administración	1	3.5	5
Velocidad de la red	1	4	5
Escalabilidad	3	3.5	5
Movilidad	2	2	5
PROMEDIO	2.875	3.5625	3.6875

V. CONCLUSIONES

- En general se concluye, en base al proyecto, que es posible la aplicación del modelo de referencia del Internet of things en la implementación de todas las plataformas propuestas Arduino, Digi e IPv6, variando el nivel de complejidad en su construcción por sus características de ser Open Hardware en el caso de Arduino, comercial para el caso de Digi y la carencia de dispositivos IPv6.

- Se puede evidenciar la gran ventaja que proporciona el manejo de datos en la nube (cloud), demostrando que no se requiere de infraestructura propia para el almacenamiento de aplicaciones ya que se puede utilizar la estructura necesaria de un proveedor, como en este proyecto se utilizó la plataforma Google para el prototipo basado en Arduino y Etherios, que es una plataforma de servicios propia de Digi.

- La monitorización de sensores y la gestión de dispositivos desde la nube, no es factible en tiempo real ya que los retardos presentes en la adquisición de datos son aleatorios, por lo tanto a pesar que se puede realizar el

procesamiento de información en la nube, no es aconsejable para aplicaciones críticas.

- El modelo de referencia del Internet of things se basa en cuatro capas, en la capa de dispositivo la plataforma Wireless Sensor Network es la base para su desarrollo, en la capa de red se emplea el protocolo IPv4 e IPv6 con el protocolo TCP como transporte, en la capa de servicio, se proveen opciones de almacenamiento y procesamiento de datos, por ´ultimo en la capa de aplicación se utiliza el esquema HTTP-REST.

- En base a las pruebas realizadas se concluye que el prototipo basado en la plataforma comercial Digi presenta menores retardos en la gestión de dispositivos debido a que el coordinador de la red Zigbee tiene mayor alcance y potencia de transmisión, posee una interfaz web propia que realiza un mejor tratamiento de los datos comparada con la plataforma Arduino en la que el coordinador es ensamblado de forma manual y su programación es más compleja.

- Se concluye que la diferencia entre las redes tradicionales IoT (Arduino y Digi) y el prototipo implementado basado en IPv6 es que en esta plataforma no es necesaria la existencia de un nodo coordinador que concentre la información de los nodos finales, debido a que cada nodo se conecta directamente con su interfaz de gestión mediante una dirección IPv6, eliminando el retardo que posee la red Zigbee y permitiendo una fácil administración de los dispositivos. Además al poseer mayor cantidad de direcciones IP, el prototipo basado en IPv6 es más escalable que los prototipos tradicionales.

- Al ser Digi un producto comercial, la implementación basada en esta plataforma es más costosa que el prototipo basado en Arduino debido a que es Open Hardware y toda su infraestructura se la construye manualmente. En el mercado se encuentran innumerables dispositivos que aplican el modelo IoT como sensores y actuadores WiFi, bombillas Zigbee-WiFi, entre otras.

BIBLIOGRAFÍA

[1] J. Ranz, The Things's City, 2013
http://thingscity.com/la-iot-segun-sap/
[2] International Telecommunication Union, Overview of the Internet of things, 2012.
[3] Google Developers, Google App Engine Tutorial, 2013
https://developers.google.com/appengine/docs/whatisgoogle appengine?hl=es
[4] C. Doukas, Building Internet of Things with the Arduino, 2012
[5] Digi International Inc., Google App Engine Device Cloud Client, 2010
http://www.digi.com/wiki/developer/index.php/Google
App Engine iDigi Client
[6] Mandat International, Universal Integration of the Internet of Things through an IPv6-based Service Oriented Architecture enabling heterogeneous components, 2012
[7] The Core Team, Tiny Core Project, 2009
http://www.tinycorelinux.net/intro.html
[8] gogo6 InC., CLIENT GUIDE, 2002
http://content.gogo6.com/gogo dc 0010 gogoclient guide.pdf
[9] D. Contero, A. Albarracín, Development of a prototype wireless sensory network implemented with Zigbee technology for monitoring water quality, 2012.
[10] Belkin International Inc., WeMo Switch, 2013
http://www.belkin.com/us/F7C027-Belkin/p/P-F7C027/
[11] N. Hardiman, IPv6 and the Internet of Things, 2013
http://www.techrepublic.com/blog/data-center/ipv6-and-the-internet-ofthings/
[12] R. Milln, El Protocolo IPv6 (I), 2006
http://www.ramonmillan.com/tutoriales/ipv6 parte1.php
[13] Ministerio de Ciencia y Tecnología de ESPAN˜A, IPv6 Servicio de Información y Soporte, 2004
http://www.6sos.org/pdf/la transición de las redes academicas:rediris v2.pdf