Las Redes Definidas por Software y los Desarrollos Sobre Esta Temática en la Escuela Politécnica Nacional

Bernal Iván[1]; Mejía David[1]

[1]*Escuela Politécnica Nacional, Facultad de Ingeniería Eléctrica y Electrónica, Quito, Ecuador*

Resumen: Este artículo aborda dos aspectos principales. El primero pretende introducir los conceptos fundamentales de las Redes Definidas por Software (SDN) y sus ventajas, revisando para ello la arquitectura básica inicial de las redes tradicionales y el estado actual de las mismas; se usan brevemente como casos de discusión: el Internet, los ISP y el *core* de los futuros sistemas celulares de quinta generación (5G). También se explican los desarrollos tecnológicos que ha permitido llegar a las SDN como la virtualización y la NFV (*Network Functions Virtualization*). Como segundo punto importante, dada la poca difusión de las SDN en el Ecuador y la gran proyección que tiene actualmente esta tecnología a nivel mundial, se describen los desarrollos en el campo de las SDN realizados por el grupo de trabajo en SDN de la Escuela Politécnica Nacional, principalmente los resultados obtenidos durante la ejecución de dos proyectos de investigación.

Palabras clave: SDN, Virtualización, NFV, Controlador SDN, Plano de Control, OpenDayLight, Trema, RYU.

Software Defined Networks and Developments in This Area at the National Polytechnic School

Abstract: This article deals with two main aspects. The first one aims to introduce the fundamentals of Software Defined Networks (SDN) and its advantages, for accomplishing that it reviews the initial basic architecture of traditional networks and their current state; the Internet, ISPs and the core of future fifth-generation cellular systems (5G) are briefly used as discussion cases. Technological developments that have enabled SDN such as virtualization and NFV (Functions Network Virtualization) are also explained. As the second important point, given the limited diffusion of SDN in Ecuador and the great future projection this technology has worldwide, the developments in the field of SDN made by the SDN working group of the National Polytechnic School are described, mainly the results obtained during the execution of two research projects.

Keywords: SDN, Virtualization, NFV, SDN Controller, Control Plane, OpenDayLight, Trema, RYU.

1. INTRODUCCIÓN

Enfocándose en el Internet, fácilmente se puede notar el incremento del número de usuarios y de servicios como los de comercio electrónico, redes sociales, servicios móviles, servicios de virtualización y cloud computing, etc. (Chico et al., 2014); frente a esto se han planteado cambios e innovaciones para satisfacer esta creciente demanda. Sin embargo, es claro que la arquitectura actual del Internet no es lo suficientemente flexible y se evidencia la necesidad de alternativas para reestructurar su arquitectura.

Redes como las de *core* de los sistemas celulares adolecen de problemas similares (4G Americas, 2014). Generalizando lo mencionado en el párrafo anterior, las arquitecturas de red tradicionales presentan limitaciones frente a nuevos y demandantes requerimientos, como por ejemplo: tienen limitada capacidad de adaptación a nuevas tecnologías, poca escalabilidad e ineficiente uso de políticas de control de acceso. Esto ha impulsado la búsqueda de nuevas alternativas a las redes tradicionales; la *Open Networking Foundation* (ONF) plantea las Redes Definidas por Software (*Software Defined Networks*, SDN) para satisfacer los nuevos requerimientos impuestos sobre las redes actuales.

Las SDN constituyen una arquitectura de red cuya característica fundamental es desacoplar físicamente el plano de control (inteligencia) del plano de datos, derivando el control a una computadora (controlador) y esperando contar con dispositivos muy rápidos en las tareas de conmutación aunque con limitada inteligencia. Esto facilita un mayor control y nivel de gestión sobre los dispositivos de red, flexibilizando el cambio de su funcionalidad y el de la red en general, empleando para ello un control centralizado, dado que el controlador tiene asociados muchos dispositivos.

Debido la poca difusión de las SDN en nuestro país y la gran proyección que tiene actualmente esta tecnología a nivel mundial, es fundamental que sea la academia la que incursione en esta temática.

ivan.bernal@epn.edu.ec.

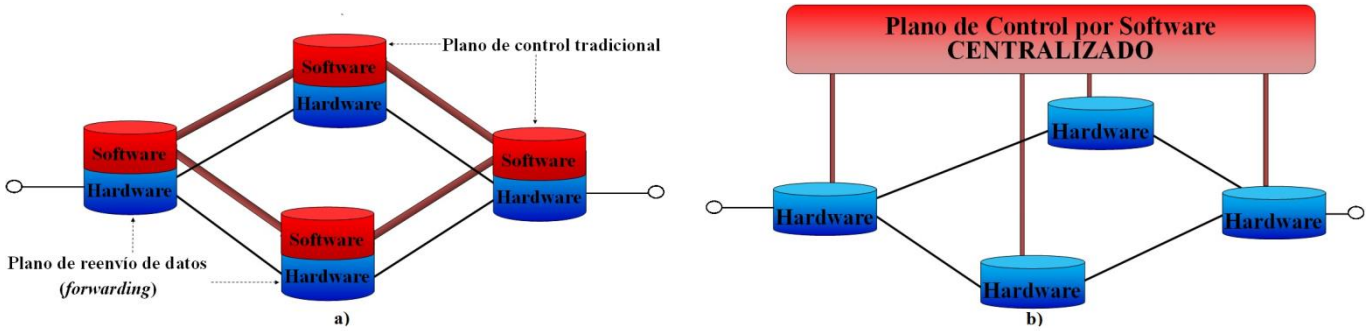

Figura 1. a) Plano de datos y control distribuidos y co-localizados en una red tradicional; b) Plano de control centralizado en una SDN (McKeown, 2014)

En la Escuela Politécnica Nacional (EPN) se inició el trabajo en las SDN por medio de proyectos de investigación y se ha difundido esta tecnología hacia otras universidades y organismos relacionados a las redes como CEDIA.

Las demás secciones de este artículo están organizadas de la siguiente manera: la Sección 2 delinea el camino que se ha recorrido hasta llegar a las SDN, revisando la arquitectura básica inicial de las redes tradicionales y el estado actual de las mismas, se explican los desarrollos tecnológicos que ha permitido llegar a las SDN y las características fundamentales de las mismas. La Sección 3 elabora sobre tres escenarios que corroboran la necesidad de una arquitectura alternativa como la de las SDN, se utiliza para esto: el Internet, los ISP y el *core* de los futuros sistemas celulares de quinta generación (5G). La Sección 4 discute la relación de las SDN con conceptos relacionados como la virtualización en general, y, en particular, el que propone virtualizar las funciones de red. La Sección 5 describe los desarrollos en el campo de las SDN realizados en la EPN, principalmente los obtenidos durante la ejecución de dos proyectos de investigación. El artículo concluye con la Sección 6 que menciona las conclusiones del trabajo.

2. EL CAMINO HACIA LAS SDN

2.1 Redes tradicionales

La estructura actual de las redes tradicionales está marcada en esencia por la presencia de *routers*. En estas redes se discriminan los planos de datos y de control, este último es distribuido.

Plano de datos: constituido por el hardware que tiene entre sus tareas el reenvío (*forwarding*) de paquetes IP hacia su destino (Figura 1a).

Plano de control: este plano está implementado en software y tiene naturaleza distribuida (Figura 1a); el software está a cargo, por ejemplo, del enrutamiento e ingeniería de tráfico. Este plano requiere que se transporte el tráfico de señalización que determina el cómo realizar el enrutamiento a través de los dispositivos de red.

2.2 Las SDN

Las SDN son una alternativa de solución a los cambios requeridos en las redes tradicionales y que serán expuestos más adelante. En esencia, las SDN son redes de dispositivos que desacoplan físicamente el plano de datos (hardware) del plano de control (software), como se indica en la Figura 1b.

Las tareas del plano de control ya no se ejecutan en el dispositivo, sino en un servidor separado denominado servidor controlador o simplemente controlador, físico o virtualizado, incluso en la nube o en *clusters* de servidores distribuidos. Un único plano de control (centralizado lógicamente) controlará a varios dispositivos cuya tarea se limita, en esencia, a realizar el reenvío de los paquetes.

El control de la red se lo realiza con la instalación, en los dispositivos de red, de reglas que definen patrones de comportamiento del tráfico. Dichas reglas se programan en el controlador empleando lenguajes de alto nivel para procesar el tráfico, definir las reglas y enviarlas a los dispositivos de red para que se apliquen sobre el tráfico que circula por ellos. Por lo tanto, la gestión de la red se encuentra altamente centralizada en el controlador que maneja la inteligencia de la red. El controlador es el encargado del manejo del plano de control y, como una posible alternativa, emplea el protocolo OpenFlow para comunicarse con los dispositivos de red (Figura 2). Pox, Ryu, Beacon y Opendaylight son ejemplos de controladores existentes al momento (Figura 2).

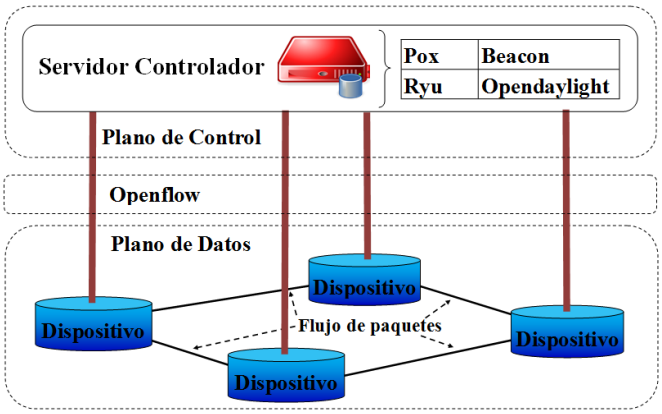

Figura 2. Plano de datos y control con el servidor controlador

2.3 Situación Actual

Las redes de telecomunicaciones actuales son estáticas y complejas e involucran una diversidad de equipos como *routers*, *switches* y *middleboxes*. El tráfico de paquetes IP es controlado por las reglas y políticas de administración y control. La administración se preocupa de fallas, configuración de los dispositivos, registro de actividades, rendimiento y seguridad.

Las *middleboxes* no son sino dispositivos de red especializados que inspeccionan, filtran, manipulan o transforman el tráfico con distintos objetivos; es decir, no se limitan al simple reenvío de paquetes. Ejemplos de *middleboxes* son los optimizadores de WAN, traductores NAT, *proxies*, detección de intrusos y sistemas de prevención, y todo tipo de *firewalls*.

La inflexibilidad de las redes de hoy hace que sea difícil tanto atender las futuras demandas de servicios que tienen naturaleza dinámica así como introducir rápidamente nuevas tecnologías innovadoras (Pretz, 2014a).

En esta situación actual debe contemplarse también, entre otros, los siguientes aspectos:

- Los avances en la tecnología.
- La disminución del costo de la conectividad con grandes anchos de banda.
- El Internet de las cosas, con sus miles de millones de nuevos dispositivos cargados de sensores.
- La disminución del costo de las Tecnologías de la Información (TI).
- La caída del precio del hardware.
- La creciente disponibilidad de software de código abierto relativamente barato.

Con los avances en las TI y "*cloud computing*", con el crecimiento exponencial del número de máquinas inteligentes conectadas a las redes de telecomunicaciones; y, con el advenimiento del Internet de las Cosas (*Internet of Things, IoT*), se espera que el tráfico en estas redes se incremente significativamente (Pretz, 2014a). Los costos del hardware están bajando y se tiene a disposición más software de código abierto. En el caso del hardware, los costos están disminuyendo, en parte, por el uso de "*whitebox switches*" y "*whitebox servers*", que son dispositivos sin marca ni características adicionales (McKeown, 2014).

Por lo mencionado, es claro que se necesita un cambio en cómo las redes se diseñan y cómo se las opera, las redes deben llegar a ser más flexibles y más fáciles de actualizar.

2.4 Las SDN y su visión

La flexibilidad de las SDN puede emplearse para resolver los problemas de las redes actuales. Un aspecto clave de las SDN es que proporcionan una serie de abstracciones y API (*Application Programming Interfaces*) para programar y controlar las funciones y los servicios de los recursos de red.

A lo largo de los años, una gran cantidad de procesamiento, almacenamiento y recursos de comunicación se han movido a lo que se llama el "borde" (*edge*) de las redes actuales. En este contexto, las SDN ayudarán a las empresas a ofrecer nuevos productos y servicios embebidos en las capacidades de comunicación (Pretz, 2013). Las SDN proporcionarán a los operadores una red muy flexible y dinámica ya que se permitirá que en sus bordes se interconecte un gran número de terminales (*smartphones*, tabletas, etc.), máquinas, cosas inteligentes y robots (Pretz, 2014a).

Las comunicaciones y procesamiento embebido se enfocan en poner todo sobre los usuarios en la nube (*cloud*) o en la neblina (*fog*), la cual extiende la computación en la nube hacia el borde de la red. Según Pretz (2014b), con la enorme cantidad de aplicaciones y servicios de software, se tendrá un gran número de máquinas inteligentes capaces de sensar, procesar e intercambiar información, además de entender lo que está sucediendo a su alrededor y adaptarse a los cambios.

La disponibilidad de software de código abierto y de hardware de TI de bajo costo y alto rendimiento, permitirá a los proveedores de servicios instalar, programar y ejecutar funciones y servicios, de igual forma que hoy lo hacen con las aplicaciones. Con el tiempo, la distinción entre "la red y la nube" y "la red y lo que se conecta a ella" desaparecerá ya que las funciones de la red podrían ser ejecutadas en la nube, en un nodo de la red, o incluso en terminales de los usuarios (Pretz, 2014a).

Las redes futuras dependerán cada vez más del software y es probable que este proceso de "softwarización" acelerará el ritmo de la innovación, facilitará nuestras vidas y ayudará a reestructurar la economía mediante la optimización de procesos (Pretz, 2013).

2.5 Inicio y evolución de las redes tradicionales

En los años 60, la genialidad de los pioneros del Internet fue mantener simple el conjunto de enlaces y *routers*; estos *routers* debían ser poco inteligentes pero rápidos, de bajo costo, fáciles de mantener y con pocas actualizaciones, ubicando en los *hosts* todo lo que se pudo de la inteligencia de la red. Estas computadoras del borde podrían actualizarse en el futuro para agregar nuevas características tales como control de congestión y seguridad pero sin tener que cambiar la red (McKeown, 2014). Todo lo mencionado constituye uno de los principios fundamentales en el que se basó la arquitectura original del Internet.

La red podría centrarse en reenviar (*forwarding*) paquetes tan rápido como fuese posible. Una red simple es más fácil de administrar y fue diseñada desde el inicio para que el control sea distribuido en lugar de centralizado, lo que constituye otro principio arquitectónico del diseño inicial del Internet.

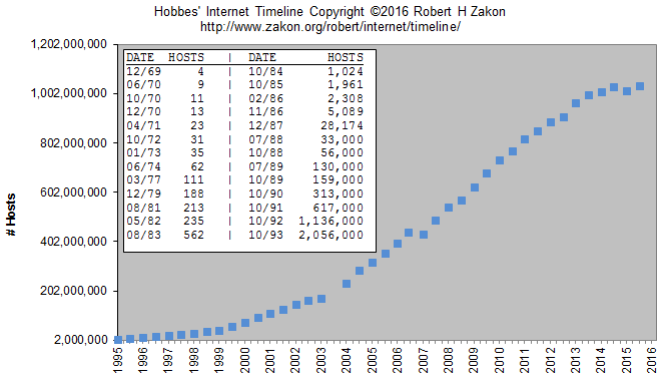

Figura 3. Crecimiento de los host conectados a Internet (Zakon, 2016)

Esta estructura simple de la red, con control distribuido, permitió su crecimiento explosivo en los 90, con el aparecimiento de muchas pequeñas empresas que ofrecieron servicios de Internet, lo que se evidencia en la Figura 3 que presenta el crecimiento exponencial del número de *hosts* conectados al Internet.

Con el tiempo, la red se hizo más grande y compleja, alejándose de la intención original, con miles de complicadas características contenidas en *routers* de naturaleza cerrada e integrados verticalmente y de un alto costo. Las redes complejas son demasiado costosas de poseer y operar.

Cada *router* y *switch* en una red es controlado en diferentes planos con el fin de manejar de manera consistente el tráfico transportado. Los planos de control deciden cómo enrutar el tráfico que fluirá luego en el plano de datos y son los planos de control en dónde están embebidas (*embedded*) las aplicaciones de red.

Las redes se hicieron más difíciles de administrar y hoy en día las redes funcionan gracias a individuos que han llegado a ser verdaderos maestros de la complejidad. Además, quienes poseían grandes redes cayeron bajo el dominio absoluto de sus proveedores de equipos. La innovación se hizo lenta, los equipos poco confiables y los márgenes de ganancias se dispararon. Ese era el estado en los 2000.

Como se mencionó, la idea inicial era que la "inteligencia de la red", que determina cómo la red debe manejar el tráfico, se reparta entre todos los dispositivos distribuidos a lo largo de un camino y se demostró que la idea era robusta para la prestación de servicios de extremo a extremo. Sin embargo, la necesidad de inteligencia y control lógicamente centralizados se evidenció cuando se trató de atender de manera personalizada muchos tipos de aplicaciones con diferentes requisitos de calidad de servicio (Pretz, 2014a).

2.6 Surgimiento de algunas ideas que sustentan las SDN

Cuando algunas compañías empezaron a construir por sí mismas sus *data centers* con miles de *switches* y *routers*, surgió la necesidad de colocar la red y su evolución bajo su propio control. En estos centros, los *firewalls* y balanceadores de carga, sobrevalorados en costo, fueron reemplazados por dispositivos con software desarrollado por personal de los propios *data centers*, ejecutándose en servidores. Los *routers* y *switches* se simplificaron, haciéndolos más confiables, de menor costo y menor consumo de energía. El resultado fue que la red completa quedó bajo el control del software, lo que puede considerarse como el origen de las SDN. A su vez, las SDN hicieron más fácil usar los servidores, el almacenamiento y la red y permitieron hacerlo de manera más eficiente, y de paso se dio origen a la denominada virtualización de redes.

Una vez presentada la breve reseña del origen de las SDN, conceptualmente se tiene que en esta nueva arquitectura de red se puede consolidar el plano de control para que un solo programa, lógicamente centralizado, controle múltiples elementos de hardware. El plano de control determina el estado de los elementos del plano de datos (*routers*, *switches* y *middleboxes*) empleando API bien definidas. Son estas API las que permiten que las aplicaciones puedan comunicarse con la infraestructura a través de un controlador SDN.

Considerando que la red es configurada y controlada por software, se proporciona a los proveedores mayor flexibilidad en la forma en que pueden utilizar los recursos de su red. Las redes de telecomunicaciones futuras se parecerán cada vez más más a un sistema de cómputo distribuido; una función de red podría ser ejecutada ya sea en la nube o en nodos de telecomunicaciones distribuidos (Pretz, 2014a).

Retomando la discusión previa sobre la visión que se tuvo en los años 60, simplemente se está regresando a una de las ideas originales, mantener simplicidad en la infraestructura conformada por enlaces y *routers;* en general, los dispositivos de red deben tener baja inteligencia pero muy rápidos en su tarea de conmutación y con el control lógicamente centralizado.

En la próxima década, se espera que muchas más redes sean definidas por software y virtualizadas. Estas redes probablemente incluirán a los grandes proveedores de servicios de Internet (AT&T, NTT, China Telecom, DT, BT, etc.), a las redes domiciliarias (para centralizar la configuración y que no sea responsabilidad del usuario conocer esos detalles), a las redes WiFi, a las redes celulares, a las redes empresariales y de los campus universitarios.

Las SDN se promocionan como una tecnología que puede desplazar a la tradicional o establecida, y es una tecnología innovadora y capaz de crear una nueva industria (*disruptive technology*). Esta revolución tecnológica cambiará las redes a través del software, no del hardware; sin dejar de lado que existen aspectos técnicos y económicos que se pueden mejorar con la introducción de las SDN.

Las SDN no son un concepto aislado, la virtualización y, en particular, la virtualización de funciones de red (NFV, *Network Functions Virtualization*) son temas íntimamente relacionados. La NFV aplica la virtualización de los CPU y otras tecnologías de computación en la nube para migrar las funciones de red del hardware dedicado a máquinas virtuales que se ejecutan en hardware de propósito general.

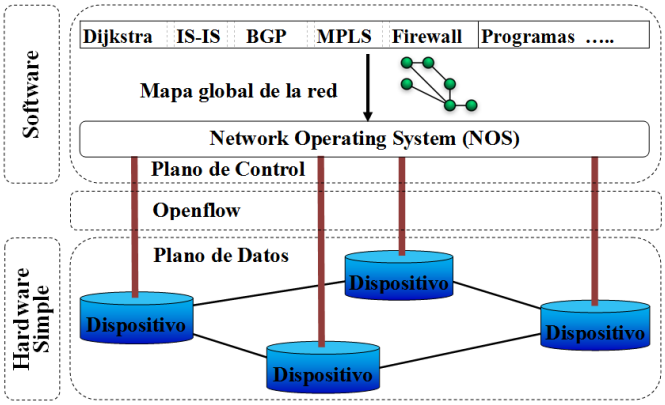

Figura 4. Obtener una vista global de la red para que la usen los diversos Programas/Aplicaciones (McKeown, 2014).

3. NECESIDAD DE LAS SDN

3.1 Se requiere un Sistema Operativo de Red

De forma simplificada, para la tarea de enrutamiento se requiere que los *routers* encaminen los paquetes al destino correcto y que tenga una vista de la estructura de la red lo más actualizada posible. Los *routers* indican a sus vecinos que están ahí y envían información sobre los enlaces que los interconectan; cada *router* debe construir un grafo de la red para luego encontrar el camino más corto (el algoritmo más famoso es el de Dijkstra, propuesto en 1956). Construir un sistema distribuido con información consistente y actualizada para una red de gran cobertura es realmente difícil.

El problema de determinar el grafo de la red es algo común para implementar varias características que se ofrecen en los equipos que forman parte del Internet; a más del Algoritmo de Dijkstra, procedimientos de MPLS, protocolos de enrutamiento como IS-IS y BGP, *firewalls*, etc., todos enfrentan el mismo reto (Figura 4).

Por otro lado, para administrar las redes actuales, en muchos casos, aún se deben realizar configuraciones de bajo nivel de los diversos componentes de forma individual y en ocasiones incluso se requiere conocer demasiados detalles sobre la red; así, para bloquear el acceso de un usuario por medio de una ACL (*Access Control List*) se debe conocer la dirección IP actual del usuario. Este escenario se parece al de una computadora sin sistema operativo en el que la configuración de los componentes que dependen de la red corresponde a tener que programar con un lenguaje de máquina que depende del hardware; de esto se desprende la necesidad de un sistema operativo de red que provea una interfaz de programación uniforme y centralizada a toda la red. De forma análoga a lo que brinda un sistema operativo en una computadora para el acceso a varios recursos para tareas de lectura y escritura, un sistema operativo de red proveería la habilidad de observar y controlar una red (Gude et al., 2008).

Entonces es razonable que el sistema operativo de red sea el responsable de construir el grafo de la red para que lo usen todos los programas y aplicaciones que lo requieran (Figura 4). El sistema operativo de red no administraría la red sino que proveería la interfaz de programación y un modelo centralizado de programación; las aplicaciones implementadas sobre este sistema operativo serían las que realicen las tareas de administración y se programarían como si toda la red estuviese en una sola máquina. Los controladores son los encargados de implementar esta visión de sistema operativo de red.

3.2 Altos y crecientes costos para operar las redes

A medida que las redes son más grandes y complejas, los gastos operacionales (*operational expense*, OPEX) de las redes crecen. Este componente de los costos totales son cada vez más significativos que los correspondientes gastos de capital (*capital expense*, CAPEX). Considerando la capacidad de las SDN de acelerar la automatización de las tareas de administración, la capacidad de implementar y ofrecer nuevos servicios de forma más rápida, y la facilidad de reasignar equipos entre diferentes servicios, se puede prever tener menores gastos operacionales (Göransson y Black, 2014). Incluso, las SDN pueden reducir los costos asociados al consumo de energía de los equipos de red, por ejemplo, en un *data center*.

3.3 ISP

Los ISP tienen un claro interés en reducir sus costos, hacer sus redes más simples, que sus equipos hagan solamente lo necesario y eliminar las características que no se usan, tomar control sobre sus redes y reducir el consumo de energía.

Según McKeown (2014), los grandes ISP afrontan un crecimiento del tráfico IP del orden del 40-50 % por año. Los usuarios finales pagan su tarifa mensual y no quieren pagar más a pesar que están generando más tráfico, es más esperan que los precios bajen. Por lo tanto, los ISP requieren que el costo de propiedad se reduzca en un 40-50 % por Gbps por año, pero en la práctica, lo reducen en menos del 20 %. La consecuencia, en un momento dado no habrán ganancias (Figura 5).

Algunas de las formas en las que las SDN pueden ayudar a mejorar los ingresos y disminuir los costos en los ISP son: a) Administración de los anchos de banda; b) Ahorros en los CAPEX y OPEX; y, c) Cumplimiento de las políticas establecidas en varios puntos de la red, incluidos los límites con otros proveedores (Göransson y Black, 2014).

3.4 Data Centers

Los *data centers* están creciendo exponencialmente en términos de tráfico, servicios y distribución geográfica. A esto se agrega la necesidad de separar el tráfico en instancias virtuales diferentes (*tenants*) para clientes o servicios del *data center*.La forma tradicional de realizar esta tarea era empleando VLANs, con el inconveniente que ésta es una solución de capa dos, que carece de inteligencia para construir algo efectivamente escalable, incluso por la limitación de los VLAN ID de 12 bits (Sanchez-Monge y Szarkowicz, 2016).

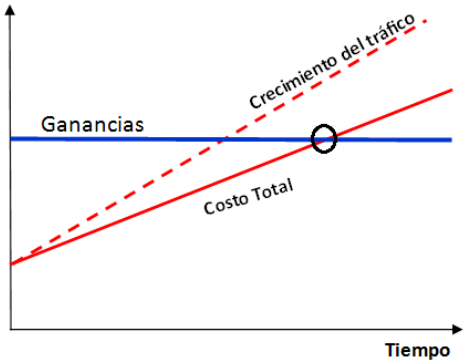

Figura 5. La situación que afrontan los ISP por incrementos de tráfico (McKeown, 2014)

Nuevas estrategias debieron ser desarrolladas, las SDN son un resultado y una alternativa de este proceso. Con las SDN se pueden introducir cambios, mejoras y nuevas características sin depender de los fabricantes/vendedores de equipos, ya que los propios *data centers* controlan el software.

Los *data centers* están atravesando grandes cambios, incluso diseñando sus propios switches, introduciendo con esto un fuerte ahorro en la gran inversión en infraestructura que deben realizar. Al hacer esto, los *data centers* tienen más control, dado que la red es más flexible, pueden también diferenciarse de otros, generando competencia y beneficios para los usuarios finales.

3.5 El core de los sistemas celulares 5G

Los requisitos y retos de los futuros sistemas móviles 5G a ser desplegados alrededor de 2020 (Huawei, 2013) incluyen: mayores tasas de datos para los usuarios, mayor número de dispositivos, mayor volumen de tráfico, menor latencia, más espectro, mayor eficiencia espectral, cobertura en interiores mejorada, bajo consumo de energía, alta fiabilidad y otros. Todo orientado a tener una sociedad móvil y conectada.

Para satisfacer estas nuevas demandas, no solo se debe enfocar en mejoras en los terminales, el interfaz de aire, las redes de acceso y *backhaul* sino en cambios radicales en el *core* de las redes celulares. Novedosas técnicas celulares como CoMP (*Coordinated MultiPoint*) pueden requerir altas tasas de datos para enviar información de control para la tomas de decisiones y envío de datos a un terminal desde varias estaciones base (Akyildiz et al., 2010).

Se han propuesto alternativas precisamente basadas en SDN y NFV. La arquitectura para el *core* de redes 5G podría ser implementada con una arquitectura SDN/NFV apuntando a la virtualización de tantas funciones como sea posible, incluidas funciones relacionadas con dispositivos, infraestructura (móvil/fijo), funciones de red y funciones de gestión.

Estas arquitecturas permitirán el desarrollo de una infraestructura altamente flexible que permitiría a los operadores la introducción y el despliegue de nuevos servicios de forma más flexible y tener un mayor ritmo de innovación (4G Americas, 2014).

3.6 Ventajas inmediatas de las SDN

- Coloca a los propietarios y operadores de red en control de su propio futuro. El software del controlador puede desarrollarse o adquirirse, usar código abierto o no, etc., una gama de opciones.
- Las redes son más fiables y más seguras puesto que propietarios y operadores son los que más conocen cómo operar su red y pueden incluir todo lo necesario para cumplir estas dos características.
- Las redes cuestan menos, el hardware es simple y racionalizado como en los inicios del Internet.
- Cuesta menos operar las redes; redes personalizadas, pueden dejar de lado las características no deseadas.

4. VIRTUALIZACIÓN, NFV Y LAS SDN

4.1 Virtualización

La virtualización normalmente se ha centrado en:

- La creación de recursos lógicos, piezas de software que se ejecutan en un servidor en hardware.
- La emulación de capacidades de hardware, como las de una CPU x86.

La novedad aquí es que las funciones de la mayoría de equipos, incluidos las *middleboxes*, podrían ser virtualizadas, incorporadas, y trasladadas a diversas ubicaciones en la red, de acuerdo a las necesidades que se presenten (Pretz, 2014a).

Las SDN y la virtualización no son conceptos nuevos, pero gracias a un mejor rendimiento y a los menores costos de hoy en día, están ya ayudando a reinventar la arquitectura de las redes y servicios, según los expertos.

Las SDN se enfocan en la creación de redes virtuales muy dinámicas en base a una variedad de nodos que se agregan a la red (*routers* con capacidades de cómputo y almacenamiento), dispositivos y elementos situados en el borde de la red, cercanos a los usuarios (Pretz, 2014a).

4.2 NFV

El concepto de NFV (*Network Functions Virtualization*) se remonta a la década de los 60, tiene como objetivo utilizar la virtualización de la CPU y otras tecnologías de computación en nube para migrar las funciones de red, de hardware dedicado a máquinas virtuales que se ejecutan en hardware de propósito general (Pretz, 2014a).

Las funciones de red virtualizadas son atractivas para los operadores de red debido a que se pueden migrar y adaptar para satisfacer las demandas actuales y al mismo tiempo aumentar la utilización de los recursos de la red y disminuir los costos de operación. Un ejemplo de una función de red a

virtualizarse es la de los dispositivos de L4 a L7, utilizados para balanceo de carga entre grupos de servidores.

Las funciones de red especializadas, como las ofrecidas por las *middleboxes*, se basan en hardware construido con propósitos normalmente cerrados y caros de mantener. Estas *middleboxes* no sólo contribuyen a la llamada "osificación de la red", lo que significa que las instalaciones son difíciles de modificar, sino que también cuesta mucho mantenerlas. Remover o incluso reducir el número de *middleboxes* con SDN y virtualización tendría varias ventajas, entre ellas están: el ahorro en los costos, mejoras en los servicios y una mayor flexibilidad (Pretz, 2013).

Las SDN y NFV son complementarias, pero no depende una de la otra. Por ejemplo, los servicios de red pueden ser desarrollados en software y ejecutados o emulados en recursos lógicos sin la necesidad de desplegar una SDN, y viceversa. Sin embargo, su combinación puede crear un entorno de recursos virtuales, interconectados por enlaces virtuales que se configuran o destruyen fácilmente para servir a múltiples aplicaciones. La unión o combinación de estos dos conceptos puede proporcionar una red más potente en términos de flexibilidad, capacidad de programación, capacidad de mover las funciones y recursos virtuales y al mismo tiempo reducir costos y permitir nuevos servicios. Las SDN y NFV harán que las infraestructuras de telecomunicaciones sean más omnipresentes (*pervasive*), flexibles y capaces de soportar todos los terminales que se conecten a las redes (Pretz, 2014a).

5. DESARROLLOS DE LAS SDN EN LA EPN

5.1 Proyecto Semilla: "Desarrollo de prototipos de redes definidas por software (SDNs - Software Defined Networks)"

Prototipo Básico para manejo de ICMP y ARP

Este proyecto permitió incursionar en la temática de SDN por primera vez en el EPN y, posiblemente, en el país en el 2012.

Inicialmente se realizó la simulación de una SDN empleando Mininet (Lantz et al., 2010), usando una topología similar al prototipo implementado posteriormente de manera física.

En el prototipo físico (Chico et al., 2014), para el plano de datos, se emplearon *switches* a los cuales se les modificó su *firmware* para soportar el protocolo OpenFlow (Figura 6). En el plano de control, los controladores definen y transmiten reglas de flujo a los *switches* usando OpenFlow.

En el proyecto se usaron cuatro de los principales controladores existentes en ese momento: NOX (Gude et al., 2008), POX (POX, 2015), Beacon (Erickson, 2013) y Floodlight (Lappas, 2013).

Para cada controlador se desarrolló un componente de software, en el lenguaje de programación determinado por el controlador, para definir la funcionalidad de los dispositivos de red, lo que permitió visualizar a la red, en su conjunto, como una plataforma programable.

En el componente desarrollado a ejecutarse en el controlador se programaron las reglas para manejar los flujos tanto para ICMP (*Internet Control Message Protocol*) así como para ARP (*Address Resolution Protocol*). Para la simulación se generó tráfico ICMP desde los *hosts* creados con Mininet usando el comando *ping*, lo cual inicialmente genera tráfico ARP. Los paquetes iniciales se derivan al controlador que determina las reglas a instalarse en los *switches*. Los resultados fueron satisfactorios y los módulos desarrollados fueron sometidos a pruebas tanto a nivel de simulación así como en el prototipo implementado, comprobándose además que las reglas se instalaron en los *switches* y efectivamente se aplicaron sobre los flujos de datos.

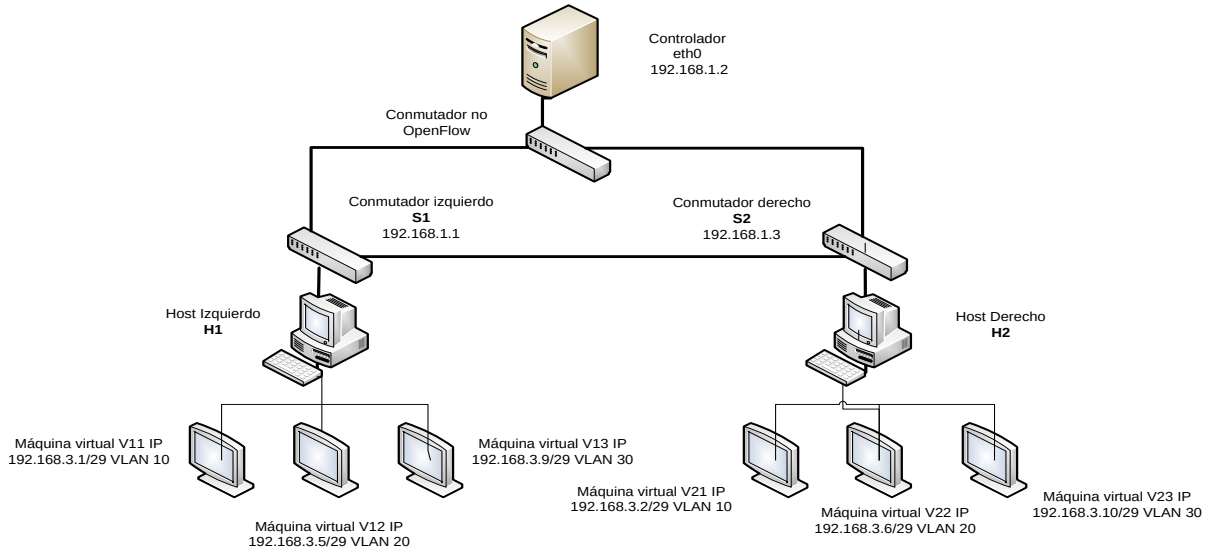

Figura 6. Diagrama de topología para la implementación del prototipo básico

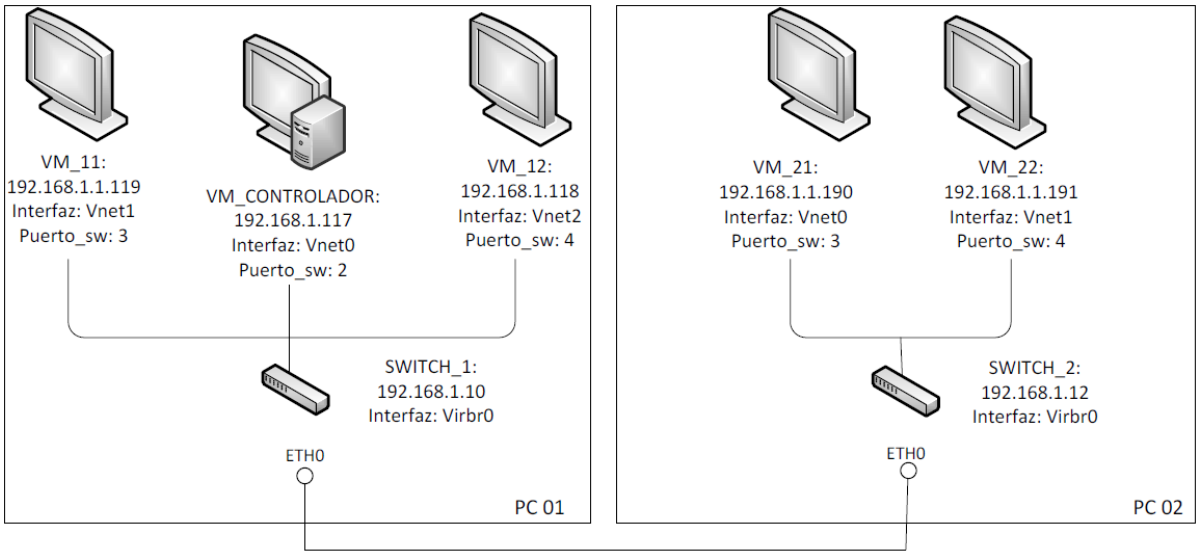

Figura 7. Diagrama para la implementación de la aplicación NAC

Aplicación NAC

Este prototipo está basado en dispositivos de software, empleando virtualización y está conformado por *switches* virtuales, un controlador y computadoras clientes; sobre este prototipo se desarrolló una aplicación para NAC (*Network Access Control*), la cual se implementó con ayuda del controlador Floodlight (Lappas, 2013) y Java.

La aplicación NAC permite controlar qué equipos tienen acceso a la red. Para esto se definen dos acciones básicas: permitir y negar; la primera se aplica cuando el equipo del cliente posee algún indicativo de seguridad, mientras que la segunda se aplica cuando se carece del mismo. Esta aplicación se basa en el modelo cliente-servidor. El servidor NAC correrá en la máquina en la que se ejecuta el controlador (Figura 7), mientras que el cliente NAC estará instalado en cada cliente.
Se empleó la API REST de FloodLight y los módulos *Firewall* y *Forwarding*, los que proporcionan las funcionalidades requeridas para el manejo de paquetes y su reenvío. El módulo *Forwarding* reenvía paquetes entre dos dispositivos. El módulo *Firewall* permite cumplir reglas ACL. El servidor controlador debe enviar el tráfico al servidor NAC para que la aplicación genere las reglas para permitir o denegar el tráfico, y una vez que tenga las reglas adecuadas, las instale en los *switches* (Morillo et al., 2014).

5.2 Proyecto Externo: "Implementación de un testbed para una SDN empleando la infraestructura de CEDIA"

El objetivo general del proyecto propuesto fue: "Implementar un *testbed* conformado por un conjunto de prototipos de SDN utilizando *switches* virtuales, habilitados y dedicados que se comunican con controladores que ejecutan módulos que definen el comportamiento de la SDN desarrollados en base a infraestructuras de software de libre distribución, y empleando la infraestructura de red de CEDIA".

Esta propuesta fue elaborada en la EPN y se invitó a participar a la Universidad Técnica de Ambato y la ESPE, esta última realizó algunas observaciones a la propuesta que finalmente fue aprobada y la dirección del proyecto la asumió la EPN. En este artículo se reportan generalidades del proyecto y sus fases así como las actividades realizadas por la EPN.

Fase I

En la fase inicial, se empleó el simulador Mininet para definir diferentes topologías de red, con diversas configuraciones de los enlaces que interconectan los *switches* así como con diversos controladores. Los controladores inicialmente evaluados por la EPN fueron OpenDayLight (ODL) (Medved et al., 2014) y Trema (Trema, n.d.).

En esta fase se desarrollaron módulos para que los *switches* se comporten como *hubs*, *switches* de capa dos, y balanceadores de carga, y otras reglas necesarias que definan lo que ocurre con el tráfico que circula por los equipos de comunicación, físicos y virtuales, utilizando lenguajes de programación comunes: Java para ODL y Ruby para Trema. Un esquema de la Fase I se presenta en la Figura 8.

A lo largo del proyecto se hizo énfasis en emplear software de libre distribución. Posteriormente, se procedió a evaluar y emplear herramientas de mayor nivel de abstracción que permitan concentrarse en la definición de la funcionalidad de la red antes que en los detalles de OpenFlow o los lenguajes de programación. En esta fase la EPN evaluó y seleccionó también a ODL. ODL hace uso de las interfaces *north bound* (NB) utilizando una alternativa REST, basada en a) HTTP para transportar los pedidos y respuestas entre clientes y servidor; y, b) XML and JSON para describir los parámetros de un pedido y de una respuesta (Medved et al., 2014).

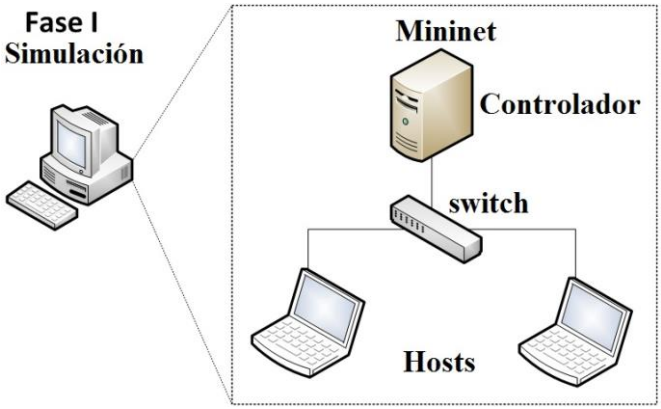

Figura 8. Esquema de la Fase I

En la etapa inicial de la Fase I, la EPN instruyó al personal de las universidades participantes sobre las ideas básicas de las SDN, su arquitectura y componentes, sobre OpenFlow y sobre el simulador Mininet (Bernal, 2015).

Fase II

Con los procesos de formación, el conocimiento adquirido y los resultados obtenidos en la Fase I, se tuvieron los insumos necesarios para llevar los módulos desarrollados, con bajos y altos niveles de abstracción, y probarlos tanto en *switches* virtuales así como en *switches* habilitados de bajo costo. Luego, se implementó un prototipo de SDN basado en *switches* virtuales y se ejecutaron los módulos desarrollados con los dos controladores seleccionados. El esquema del prototipo con *switches* virtuales se presenta en la Figura 9a. Posteriormente se procedió a cambiar el *firmware* de *switches* de bajo costo para habilitarlos para que soporten OpenFlow. El esquema del prototipo SDN usando *switches* habilitados se presenta en la Figura 9b.

Fase III

En la tercera fase se implementó el prototipo final, parte fundamental del *testbed*, con componentes de la red distribuidos geográficamente, un controlador remoto y empleando una estructura híbrida de *switches* virtuales, habilitados y dedicados, empleando la infraestructura de CEDIA para la interconexión hacia el controlador. En este prototipo se evaluó el impacto de tener el controlador remoto en la operación de la SDN. En la Figura 10 se presenta un esquema de este prototipo parte del *testbed*.

6. RESULTADOS

Los prototipos estructurados se basaron en *switches* virtuales y de bajo costo habilitados para que soporten OpenFlow así como *switches* virtuales pero también se adquirieron equipos HP E3500 modelo J9470A, con lo que se tuvo al alcance equipos con soporte nativo de OpenFlow.

A más de los distintos prototipos estructurados, se debe hacer referencia a los módulos desarrollados y a los eventos de difusión que la EPN ha organizado y otros en los que ha participado, se dictaron seminarios y conferencias en varias ciudades del Ecuador y en Colombia (Bernal, 2015).

La EPN ha trabajado sobre módulos para "*hub*" y "*learner switch*" para el controlador OpenDayLight y para Trema. Para el controlador Trema se desarrolló también un balanceador de carga. A más de lo realizado con OpenDayLight y Trema, la EPN por medio de proyectos de titulación de pregrado está concluyendo el desarrollado de un balanceador de carga empleando el controlador RYU (RYU Project Team, 2014) y la "Implementación de una Honeynet usando una SDN" empleando Pyretic (Reich, 2013).

7. CONCLUSIONES

Los prototipos y módulos desarrollados han permitido evidenciar los beneficios de las SDN y, en particular, la capacidad que brindan para modificar la red según las necesidades, visualizando a la red como una plataforma programable.

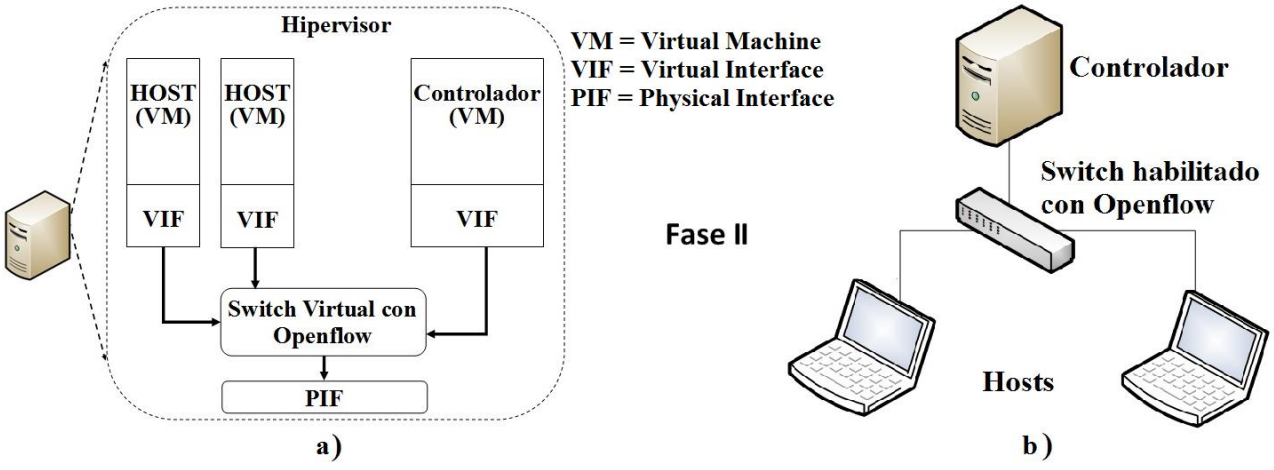

Figura 9. a) Fase II: Prototipo usando switches virtuales; b) Fase II: Prototipo usando switches habilitados con OpenFlow

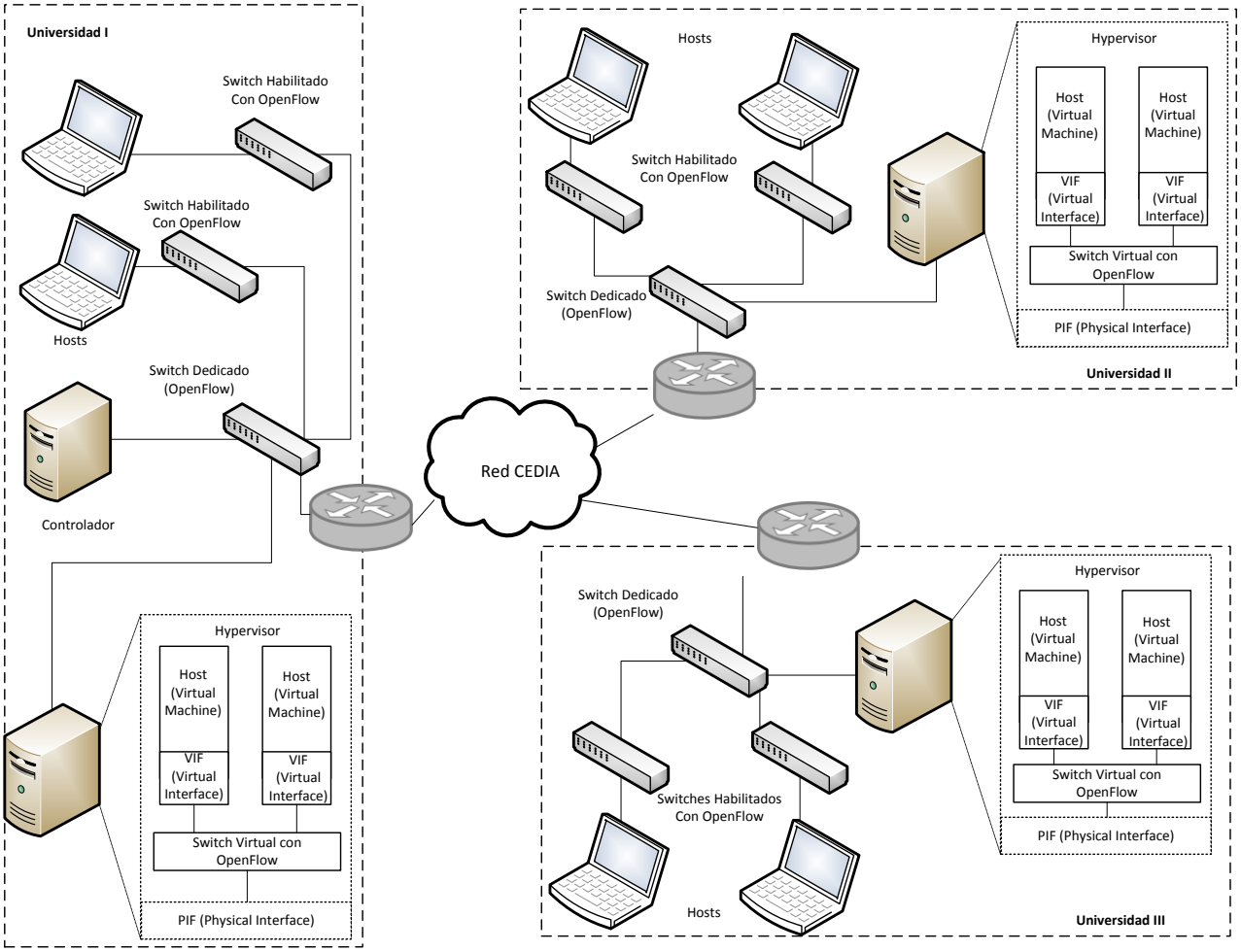

Figura 10. Esquema del prototipo híbrido del *testbed* con componentes geográficamente distribuidos

La elección del controlador es una decisión esencial al momento de estructurar la SDN y varios parámetros deben considerarse. Los módulos se desarrollaron empleando diferentes controladores que a su vez tienen API y facilidades diferentes y que requieren emplear diversos lenguajes de programación.

El conjunto de prototipos implementados y que en su conjunto conforman el *testbed* contienen una infraestructura básica que podrá ampliarse y mejorarse, y a futuro servirá para el desarrollo y ejecución de diversos módulos que definirán el comportamiento de los *switches* y de la red en sí misma.

Cabe resaltar la utilidad de la herramienta de simulación Mininet, cuando no se tiene una infraestructura de pruebas, aunque no es recomendada para verificar los tiempos de respuesta del controlador, porque al no existir retardos ocasionados por conexiones físicas, los tiempos de transmisión no se ajustan a la realidad.

Con el conocimiento adquirido en base a los dos proyectos desarrollados, se están impartiendo seminarios y materias en la EPN que cubren estos tópicos, con un adecuado componente práctico.

Las SDN no se han difundido aún en el Ecuador, ni en el ámbito académico ni empresarial al nivel como se lo ha hecho en otros países; así la Universidad de Stanford es muy activa y empresas como Google las emplean a gran escala. Es por tanto pertinente que en nuestro país se incursione en esta temática relativamente nueva y que aún está en desarrollo pero que tiene una proyección futura magnífica.

RECONOCIMIENTO

Los autores desean expresar el reconocimiento a la Escuela Politécnica Nacional y a CEDIA por el soporte financiero y logístico durante el desarrollo de las actividades de los proyectos que han permitido incursionar en la temática de las SDN. De igual manera, un reconocimiento a los estudiantes que han participado en la ejecución de los proyectos relacionados a SDN.

REFERENCIAS

4G Americas. (2014). *Bringing Network Function Virtualization to LTE*. *4G Americas*. Obtenido de: http://www.4gamericas.org/index.php/download_file/view/540/847. (Enero, 2016).

Akyildiz, I., Gutierrez-Estevez, D. M., y Chavarria, E. (2010). The evolution to 4G cellular systems: LTE-Advanced. *Physical Communication*, 3(4), 217-244.

Bernal, I. (2015). Informe final del proyecto CEPRAVII-2013-04.

Bob Lantz, B., Heller, B. & McKeown, N. (2010). A Network in a Laptop: Rapid Prototyping for Software-Defined Networks. Hotnets-IX *Proceedings of the 9th ACM SIGCOMM Workshop on Hot Topics in Networks*, 9 (1), Article No. 19.

Chico, J., Mejía, D. & Bernal, I. (2014). Implementación de un prototipo de una Red Definida por Software (SDN) empleando conmutadores habilitados. *Jornadas en Ingeniería Eléctrica y Electrónica*, 25(1), 356-365.

Erickson, D. (2013). The Beacon OpenFlow Controller. *HotSDN'13 Proceedings of the second ACM SIGCOMM workshop on Hot topics in Software Defined Networking*, 2(1), 13-18.

Göransson, P., y Black, C. (2014). *Software Defined Networks: A Comprehensive Approach*. Waltham, MA, USA: Morgan Kaufmann.

Gude, N., Koponen, T., Pettit, J., Pfaff, B., Casado, M., McKeown, N. & Shenker, S. (2008). NOX: Towards an Operating System for Networks. *ACM SIGCOMM Computer Communication Review Archive*, 38(3), 105-110.

Huawei. (2013). *5G: A Technology Vision*, *Huawei Technologies*, Version No.: M3-023985-20131104-C-1.0. Obtenido de: http://www.huawei.com/5gwhitepaper. (Enero, 2016).

IEEE. Special Report Software-Defined Networks. Obtenido de: http://theinstitute.ieee.org/static/special-report-software-defined-networks. (Enero, 2016).

Lappas, P. (2013). Introducing Project Floodlight. *Project Floodlight*. Obtenido de: http://www.projectfloodlight.org/blog/2013/03/25/introducing-project-floodlight. (Enero, 2016).

McKeown, N. (2014). IET Appleton Lecture: Software Defined Networks and the maturing of the Internet. Obtenido de: https://tv.theiet.org/?videoid=5447. (Enero, 2016).

Medved, J., Tkacik, A., Varga, R. & Gray, K. (2014). OpenDaylight: Towards a Model-Driven SDN Controller Architecture. *IEEE 15th International Symposium on a World of Wireless, Mobile and Multimedia Networks (WoWMoM)*, 15(1), 1-6.

Morillo, G., Mejía, D. & Bernal, I. (2014). Aplicación para control de acceso a la red (NAC) utilizando SDN. *Revista Politécnica*, 34(2), 27-33.

POX. POX Wiki. (2015). Obtenido de: https://openflow.stanford.edu/display/ONL/POX+Wiki. (Enero, 2016).

Pretz, K. (2013). Software-Defined Networks Explained. *The Institute: The IEEE News Source*. Obtenido de: http://theinstitute.ieee.org/benefits/ieee-groups/softwaredefined-networks-explained. (Enero, 2016).

Pretz, K. (2014a). Reinventing Communications Networks. *The Institute: The IEEE News Source*. Obtenido de: http://theinstitute.ieee.org/technology-focus/technology-topic/reinventing-communications-networks. (Enero, 2016).

Pretz, K. (2014b). Software already defines our lives but the impact of SDN will go beyond networking alone. *The Institute: The IEEE News Source*. Obtenido de: http://theinstitute.ieee.org/technology-focus/technology-topic/software-already-defines-our-lives. (Enero, 2016).

Reich, J., Monsanto, C., Foster, N., Rexford, J., & Walker, D. (2013). Modular SDN Programming with Pyretic. ;Login: The USENIX Magazine, 38 (5), 128-134.

RYU Project Team. (2014). RYU SDN Framework. Obtenido de: https://osrg.github.io/ryu-book/en/Ryubook.pdf. (Enero, 2016).

Sanchez-Monge, A., and Szarkowicz, G. K. (2016). *MPLS in the SDN Era*. Sebastopol, CA, USA: O'Reilly Media, Inc.

Trema (n.d.). Trema: Full-Stack OpenFlow Framework in Ruby and C. Obtenido de: http://trema.github.io/trema. (Enero, 2016).

Zakon, R. H. (2016). Hobbes' Internet Timeline 23. Obtenido de: http://www.zakon.org/robert/internet/timeline. (Enero, 2016).