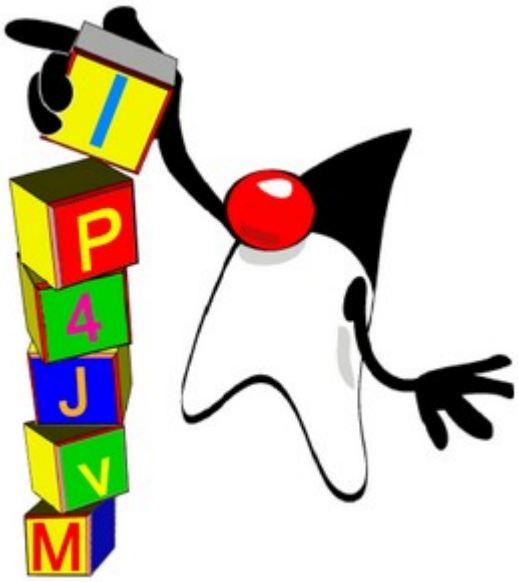


Proyecto de Grado 2008

IP4JVM

Autores:

Leandro Scasso

Marcos Techera

Tutor:

Ariel Sabiguero

Tribunal:

Andrés Aguirre

Eduardo Grampín

Carlos Martínez

Resumen

El presente informe contiene el detalle del desarrollo del proyecto de grado realizado por Leandro Scasso y Marcos Techera, el cual consistió en mejorar y completar un prototipo ya existente que agrega el manejo de stacks de protocolos a la máquina virtual Java.

El proyecto IP4JVM consiste en un framework enteramente programado en lenguaje Java, el cual implementa un stack de protocolos y realiza la comunicación con el dispositivo físico de red por medio de una implementación realizada en JNI. Mediante la integración del mencionado stack con una máquina virtual Java se obtuvo una plataforma capaz de interpretar y ejecutar código Java pero que a su vez prescinde de las funcionalidades de red brindadas por el sistema operativo y utiliza las implementadas por el stack.

Durante el proyecto se incorporó al stack el soporte del protocolo DHCPv6, así como también la capacidad de operar como router entre diferentes interfaces. Otra funcionalidad agregada sobre el router es la capacidad de realizar NAT IPv6 a IPv6. Otros objetivos desarrollados fueron la construcción de un aplicativo web que permite la configuración del stack de forma dinámica. Asimismo, se estudio la posibilidad de que el proyecto pueda correr en un dispositivo móvil.

Todas las funcionalidades, las ya existentes y las agregadas durante el proyecto, fueron probadas directa o indirectamente por test exhaustivos desarrollados por el proyecto TAHI. Estos test realizados son los mismos que realiza cualquier empresa internacional que desee certificar un producto IPv6 con el IPv6 Ready Logo.

Palabras Claves: Networking en Java. JNI. Framework. Modelos de Capas de Red. Stack de Protocolos. OpenJDK. DHCPv6. Router. NAT66. Java ME. TAHI. IPv6 Ready Logo.

Tabla de contenidos

1.Introducción

El presente documento contiene la descripción general del proyecto de grado realizado por Leandro Scasso y Marcos Techera, el cual consiste en la continuación de un proyecto de grado ya existente realizado por Roger Abelenda e Ignacio Corrales durante año 2007 [IP4JVM] que a su vez era la continuación del trabajo realizado por Laura Rodríguez.

El proyecto IP4JVM consiste en un framework enteramente programado en lenguaje Java, el cual implementa un stack de protocolos y realiza la comunicación con el dispositivo físico de red por medio una implementación realizada en JNI.

El objetivo del presente proyecto es poder incluir al stack la posibilidad de realizar el manejo de la configuración de las direccciones IPv6 mediante DHCPv6, así como también poder realizar el ruteo de paquetes y hacer uso de una solución NAT IPv6 a IPv6. Otros de los objetivos planteados es la realización de una aplicación web que permita realizar las configuraciones del framework y la realización de un estudio con motivo de poder dilucidar que tan lejos o cerca se encuentra la posibilidad de realizar el deploy del stack implementado en un dispositivo móvil. Además en todo momento se tiene como objetivo la mejora de la calidad del proyecto.

1.1.Presentación de IP4JVM

IP4JVM consiste en un stack de protocolos programado en Java, el cual mediante la integración del mismo a una máquina virtual Java se obtiene una plataforma que es capaz de interpretar y ejecutar código Java que prescinde de las funcionalidades de red brindadas por el sistema operativo para hacer uso de las brindadas por el stack del proyecto.

A parte del framework comentando anteriormente también se encuentran implementados protocolos de distintas capas, los cuales permiten la comunicación de dos nodos a nivel de capa de aplicaciones. El protocolo de enlace de datos implementado es Ethernet II[ETHII] mientras que IPv6 [RFC2460] e ICMPv6[RFC4443] son los de la capa de red. Para la capa de transporte se encuentran implementados TCP[RFC793] y UDP [RFC768].

1.2.Publico objetivo

Tanto este documento como sus anexos están orientados a personas con un perfil informático con conocimientos de redes de computadoras y su organización jerárquica en capas. En general es recomendable que el lector tenga conocimientos del stack IPv6, del protocolo DHCPv6 [RFC3315], de routeo de

paquetes y NAT. También es necesario tener una idea general del funcionamiento de una máquina virtual Java.

El lector también podría estar interesado en la lectura, previa, de los RFCs que detallan en mayor profundidad los protocolos IPv6, DHCPv6, TCP, UDP y NAT. Si bien esto no es un requisito excluyente ya que el lector podrá encontrar los conceptos básicos e imprescindibles necesarios para el entendimiento del trabajo realizado en esta documentación, si pueden aportar un mayor conocimiento de los temas tratados brindándole al lector una mayor capacidad de abstracción y objetividad.

1.3.Organización de la Documentación

La documentación del proyecto esta compuesta por un documento principal, el presente documento, y siete anexos que explican en profundidad cada una de las etapas más relevantes del proyecto así como la terminología del mismo.

En los siguientes puntos se puede obtener una noción básica de los temas que tratan cada uno de los documentos que componen la documentación del proyecto.

1.3.1.Documentación General

Se trata del actual documento, en el mismo se presenta un a introducción al proyecto así como los objetivos del mismo. Se presenta el cronograma general del mismo, una introducción al estudio del arte realizado sobre los distintos temas, el cual es profundizado en cada uno de los anexos correspondientes. Por último se presentan cuales son los resultados obtenidos, las conclusiones y los posibles trabajos que se pueden realizar sobre el proyecto.

1.3.2.Anexo I - DHCP - IP4JVM

Es el documento en donde se explica en profundidad el mecanismo de autoconfiguración de un nodo IPv6. Se presenta un amplio estudio sobre la configuración DHCPv6 y de la solución que se implementó para que el proyecto pudiera hacer uso de la mencionada configuración. También se detallan con mayor profundidad los tiempos insumidos durante esta etapa, los cuales fueron presentados de forma introductoria documento principal.

1.3.3.Anexo II – Router & NAT - IP4JVM

En este anexo es en donde es presentado de forma completa y extensa el manejo de Routing e IPv6 a IPv6 NAT y de como se llego a la solución que permite que el proyecto haga uso de los mismos. Al igual que con el anexo I, también se presentan con mayor detalle los tiempos insumidos durante esta etapa.

1.3.4.Anexo III – Aplicación web

Aquí es en donde se presenta la aplicación web que se implementó con el objetivo de poder realizar la configuración y testing del proyecto. Además se muestran los estudios realizados para que la misma pueda correr en un sistema que no posea soporte nativo para IPv6.

1.3.5.Anexo IV – Dispositivos Móviles

Este documento presenta las investigaciones realizadas sobre dispositivos móviles, sus sistemas operativos y Java Micro Edition, con el fin de poder dilucidar si es factible que el proyecto IP4JVM pueda correr en un dispositivo móvil.

1.3.6.Anexo V – Configuración de un Ambiente de Testing en IP4JVM

En este documento se presentan las herramientas que fueron utilizadas durante el transcurso del proyecto con el fin de realizar las implementaciones y tests sobre el mismo. Estas herramientas no sólo son presentadas en forma teórica, si no que para las más relevantes también son adjuntas configuraciones de ejemplo para brindar al lector una mejor idea de lo que permiten realizar las mismas.

1.3.7.Anexo VI - Testing en IP4JVM

A lo largo de este anexo se podrán ver los distintos tests que se fueron realizando durante el transcurso del proyecto, así como las configuraciones que permiten llevar acabo los mismos. Se explica el objetivo de los mismos y se muestran los resultados obtenidos.

1.3.8.Anexo VII - Glosario

Se presentan las definiciones de los distintos términos usados en todo el conjunto de documentos del proyecto.

1.4.Organización de este documento

El presente documento se encuentra organizado en capítulos, cada uno de los cuales presenta un aspecto funcional del desarrollo del proyecto.

Capítulo 1, Introducción: en la introducción se definen los aspectos generales del proyecto, tratando de brindarle al usuario un panorama general del proyecto.

Capítulo 2, Motivación y objetivos: en este capítulo son presentados los objetivos que motivaron este proyecto.

Capitulo 3, Estado del Arte: en este capítulo se presenta de forma general los estudios e investigaciones que se llevaron acabo para poder afrontar el proyecto. Este punto es extendido en los anexos.

Capítulo 4, Cronogramas y organización del trabajo: en este capítulo se presentan la distribución de las horas del proyecto en las distintas tareas que fueron realizadas en el transcurso del mismo.

Capítulo 5, Resultados y Conclusiones: aquí es donde se presentan las conclusiones del proyecto, las limitaciones del mismo y las posibles mejoras a realizarle.

Capitulo 6, Referencias: en este punto son presentadas las fuentes de información a partir de las cuales se obtuvo la información para realizar las investigaciones.

2.Motivación y Objetivos

El lenguaje Java no brinda servicios de networking de bajo nivel de forma nativa. Para resolver esto, la máquina virtual Java [JVM] utiliza los servicios que provee el sistema operativo huésped. De esta forma al programar en Java se podrá hacer uso de las clases provistas por el paquete java.net para poder comunicarse con la capa de transporte de forma transparente ya que será la máquina virtual Java quien se haga cargo de efectuar los llamados al sistema operativo necesarios para establecer una comunicación. Los problemas aparecen cuando, a través de código Java, se quiere acceder directamente los paquetes que llegan desde la capa física o poder tomar decisiones sobre como fragmentar, como y cuando enviar, en resumen cuando se quiere programar en la capa de red o en la capa de enlace.

Es por lo mencionado en el punto anterior que el lenguaje Java encuentra una limitación. Limitación para la cual no provee herramientas para solucionarla, algo que si hace el proyecto IP4JVM. Si bien el proyecto provee de un stack para hacer el manejo de los paquetes provenientes de la capa física, hay muchos de estos que aún no sabe interpretar y manejar, por lo cual se terminan descartando. En la búsqueda de tener una herramienta más completa que permita el manejo de más servicios de red es que comienzan las tareas de mejora y extensión de las funcionalidades del proyecto IP4JVM.

2.1.Objetivos

Las primeras reuniones de trabajo tuvieron el objetivo de plantear y convenir los objetivos que se buscarían cumplir durante el transcurso del proyecto. Estos objetivos intentaron continuar la línea de los objetivos planteados por los anteriores trabajos académicos que fueron quienes concibieron al proyecto IP4JVM y a su vez ser concisos y realistas, de forma que estos fueran realizables en el tiempo dispuesto para la actividad académica de proyecto de grado de la carrera Ingeniería en Computación.

Los principales objetivos que se plantearon fueron que un nodo pudiera configurarse por medio del protocolo DHCPv6, que este pudiera rutear paquetes dentro de una red y que pudiera realizar NAT/PAT IPv6 a IPv6 de una red a otra. Todas estas funcionalidades requerían ser testeadas con lo cual dentro de los objetivos del proyecto se incluyo el estudio de herramientas que permitieran realizar tests sobre el proyecto y hacer uso de las mismas. A su vez estar realizando pruebas sobre el proyecto implicaba poder realizar modificaciones de la configuración del mismo de una forma rápida y amena, es por eso que también se incluyo la implementación de una aplicación web que permitiera configurar y ayudara a testear el proyecto. Como objetivos adicionales se incluyeron dos preguntas sencillas pero que requerían de un estudio y análisis detallado para

poder ser respondidas, estás son: el proyecto IP4JVM, ¿puede ser implantado en un entorno sin soporte para IPv6? y ¿es factible realizar un deploy en un dispositivo móvil?

2.1.1.DHCPv6

Como ya se mencionó los objetivos planteados son varios pero en este punto el enfoque es sobre el objetivo que plantea la inclusión al proyecto de la posibilidad de que un nodo haga uso de la configuración DHCPv6.

Para poder realizar la implementación que permitiera lo antes mencionado se debió realizar un estudio de como se lleva acabo la autoconfiguración de los nodos IPv6 [RFC4862] y en que consiste una configuración DHCPv6 [RFC3315]. Con estas investigaciones surgió un objetivo secundario que consistía en realizar una solución cuyo diseño fuera orientado a objetos y que permitiera una sencilla incorporación de los protocolos que se configuran a partir de DHCPv6 (DNS, Dominio, NTP, entre otros).

Este objetivo se encuentra muy vinculado con el objetivo de testing y de la aplicación web, con lo cual también se planteo como objetivo secundario la necesidad de que las configuraciones del proyecto correspondientes al componente DHCPv6 se pudieran realizar de una forma sencilla y mediante invocaciones a operaciones al sistema desarrollado.

2.1.2.Routing & NAT

La búsqueda del cumplimiento de este objetivo se inició luego de culminado el desarrollo vinculado al objetivo de DHCPv6 y con este primero se buscó que un nodo pudiera realizar el routing de paquetes para luego poder realizar funciones de NAT para IPv6 entre dos redes.

Para poder alcanzar este objetivo fue necesario la investigación del funcionamiento de ICMPv6 Redirect [RFC4861] y del estudio de alternativas para la implementación de NAT para IPv6.

Al igual que el objetivo anterior este objetivo se encuentra muy vinculado al de testing y del aplicativo web, con lo cual también es necesario que las configuraciones del proyecto se pudieran realizar de forma sencilla por medio de invocaciones al sistema.

2.1.3.Testing

Como se ha mencionando previamente en este capítulo el testing de la solución desarrollada era algo primordial dentro de los objetivos del proyecto. Las herramientas que se estudiaron y utilizaron fueron varias, pero la mayor dedicación y esfuerzo estuvo puesto sobre el proyecto TAHI [TAHI]. El cual permitió obtener tests de gran utilidad para el proyecto sin la necesidad de realizar el diseño de los mismos y permitiendo ver la interoperabilidad y conformidad de lo implementado.

2.1.4.Aplicación web

Este objetivo parte de la necesidad de poder interactuar con el stack del proyecto mientras este se encuentra en funcionamiento, algo que facilita mucho la realización de los tests que se corren sobre el proyecto.

Otros de los objetivos incluidos dentro del desarrollo de la aplicación web es el de contar con aplicaciones propias, como ser el ping, para poder realizar tests propios desde el proyecto.

Aprovechando la construcción de una aplicación que corre sobre IP4JVM es que se decidió incluir en este objetivo el estudio de la posibilidad de que una aplicación que corra sobre el mencionado proyecto, como es el servidor Tomcat[TCAT], lo haga en un entorno en donde el sistema operativo no provea soporte nativo para IPv6.

2.1.5.Dispositivos Móviles

Dentro de los objetivos adicionales se planteó la incógnita de si era o no posible realizar un deploy del stack en un dispositivo móvil. Esta pregunta abarcaba muchos aspectos con lo cual la misma se subdividió en objetivos más pequeños y concretos. Estos objetivos fueron el estudio de Java Micro Edition[JME], el estudio de un grupo representativo de los sistemas operativos disponibles para dispositivos móviles y las alternativas que estos proveen para poder comunicarse con la capa física de forma de poder realizar el manejo de la capa de red.

3.Estado del Arte

En esta sección se presentan datos acerca del proyecto que precedió a este y el estado en el que se encontraba al inciar el trabajo. Esta información es de gran importancia ya que el trabajo realizado anteriormente es el punto de partida para nuestro proyecto.

También se realizara una introducción a los principales temas estudiados durante el proyecto de forma de brindarle una idea general al lector de los temas a tratar en los distintos anexos.

3.1.Proyectos Anteriores

Como se mencionó en el capítulo 1 este proyecto comienza como la continuación del trabajo realizado por Roger Abelenda e Ignacio Corrales en el contexto de su proyecto de grado. Este trabajo a su vez fue la continuación del trabajo realizado por Laura Rodríguez. Los objetivos iniciales consistían en el desarrollo de un framework Java mediante el cual se pudiera implementar un stack de protocolos. El mismo sería multiprotocolo permitiendo agregar implementaciones de distintos protocolos, todos ellos programados en Java. En una primera instancia se implementaron parte de los protocolos IPv6 [RFC2460], Ethernet II [ETHII], ICMPv6 [RFC4443] y UDP [RFC768].

En una segunda instancia, durante el trabajo de Abelenda y Corrales, se agregaron nuevos objetivos, pero manteniendo el espíritu inicial de proyecto. Como parte de estos nuevos objetivos se encontraba completar las implementaciones de IPv6, Ethernet II, ICMPv6, UDP y la de realizar la implementación del protocolo TCP [RFC793].

3.2.DHCPv6

Antes de comenzar a entender el funcionamiento de DHCPv6[RFC3315] era de vital importancia entender el funcionamiento de la autoconfiguración de direcciones en IPv6 [RFC4862] y por que este no era suficiente provocando la necesidad de utilizar DHCPv6.

A su vez para poder entender como funciona la autoconfiguración de direcciones es necesario entender los estados en los que se puede encontrar una dirección IPv6 y como es que se puede arribar a los mismos.

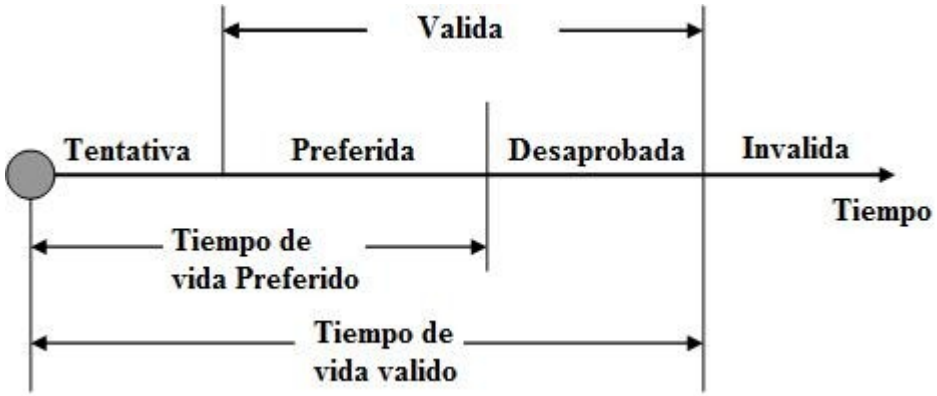


Ilustración 1: Estado de las direcciones IPv6 durante su configuración

En la figura 1 se pueden apreciar los estados por lo cuales puede pasar una dirección IPv6, cuando la dirección se encuentra tentativa es cuando se está verificando la unicidad de la misma, cuando la dirección se encuentra en este estado no puede recibir mensajes unicast. Sin embargo si puede procesar, realizar envío y recepción de ciertos mensajes multicast. Luego de poder comprobar la unicidad de la dirección esta pasa a un estado válido para luego pasar a un estado inválido. En este último estado la dirección ya no puede ni enviar ni recibir tráfico. El estado válido a su vez esta subdividido en dos estados, un estado Preferido (del inglés preferred) y un estado Desaprobado (del inglés Deprecated), la diferencia de estos dos estados es que en el primero se pueden realizar cualquier tipo de comunicación mientras que en el segundo se aconseja no establecer nuevas comunicaciones y sólo mantener las ya establecidas sin ningún tipo de restricción.

Ya se pudo apreciar los estados por los cuales puede pasar una dirección, y ahora la pregunta a responder sería, ¿cómo se inicia todo este proceso?

Todo inicia al habilitar una interfaz, a partir de este momento trata de realizar la configuración de su dirección link-local, la cual construye a partir de un algoritmo el cual sólo toma información de la interfaz a la cual se le va a asignar la dirección, este algoritmo no es objeto de estudio durante este proyecto con lo cual no ahondaremos mucho más en él. Luego de construida esta dirección el nodo comienza a dar aviso a sus vecinos de su dirección y en caso de no obtener una o más respuestas indicando que ya se encuentra siendo utilizada la mencionada dirección, esa será la dirección que se configurará como la dirección link-local. Luego de culminar la configuración de la dirección local se comienza con la búsqueda de routers dentro de los enlaces que tiene disponible el dispositivo. En caso de encontrar routers dentro de estos enlaces los mismos enviaran información al nodo indicandole que pasos debe de seguir, dentro de esos pasos de encuentra la configuración de direcciones globales a partir de un prefijo determinado por el router.

Luego de configuradas las direcciones globales, que puede ser ninguna, es que se termina lo que se conoce como "IPv6 Address Autoconfiguration", pero como se pudo apreciar durante esta configuración nunca se asignó una dirección IPv6 especifica a un host, lo más cercano es la asignación de una dirección a partir de un prefijo, ni tampoco se le asignó un dominio o un servidor de DNS o NTP. Para todo esto es que existe la configuración DHCPv6 la cual inicia con el mismo mensaje que le indica al nodo si debía de realizar o no la configuración de una dirección global.

Los routers pueden indicar tres cosas con respecto a la configuración que debe de seguir un nodo, sólo quedarnos con la autoconfiguración de direcciones, sólo obtener información adicional(todo menos asignación de direcciones IPv6) de un servidor DHCPv6 o bien obtener toda la información posible de un servidor DHCPv6. Estas configuraciones reciben los nombre de Stateless Configuration, Stateless DHCP Configuration y Stateful Configuration.

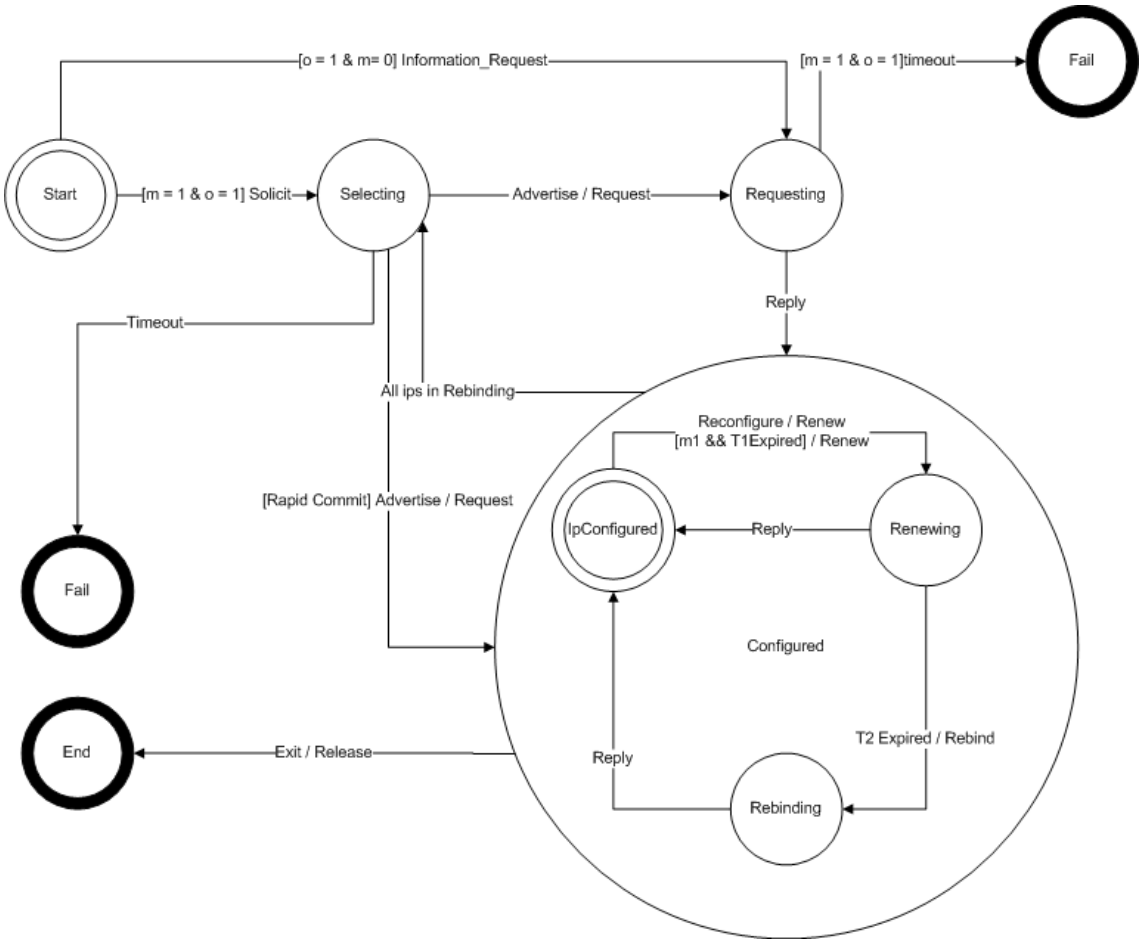


Ilustración 2: Estados de una configuración DHCPv6

Como se puede apreciar en la ilustración 2 una configuración DHCPv6 pasa por varios estados los cuales incluyen la selección de un servidor DHCPv6 para luego pasar a pedirle información al mismo y luego de configurada esta información se queda en un estado en donde la misma es renovada de forma periódica.

3.3.Router & NAT

Un router, como se define en la terminología del RFC 4861[RFC4861], es un nodo que realiza el forwarding de paquetes no explícitamente dirigidos hacia él. Mientras que un host es definido como cualquier nodo que no es un router.

Dentro de las funcionalidades de un router sobre la que se tuvo mayor interés en estudiar e implementar fue el ICMPv6 Redirect, el cual a diferencia de ICMPv4 es un mensaje de información y no de error. Estos mensajes son usados por los routers para dar aviso a los hosts de que existe una mejor ruta para un destino cuando un paquete es enviado.

Como ya se ha mencionado al iniciar el proyecto este contaba con una implementación de IPv6, TCP, UDP e ICMPv6, a partir de estos protocolos es que el proyecto brinda funcionalidades como el envió/recepción de paquetes desde un origen a un destino pertenecientes al stack de direcciones. Por lo cual sólo brinda funcionalidades para configurar hosts, pero ningún soporte para routers.

Se realizo un estudio del algoritmo de selección de dirección de IP de origen usado en el proyecto y se llego a la conclusión de que el mismo no era correcto. Esto se debía a que sólo tenía en cuenta que el scope de origen sea igual al de destino, a partir de esto se seleccionaba cualquier IP que cumpliera esta condición.

Además se detectó que no se estaban generando los prefijos de las direcciones link-local, así como, si bien se permitía la configuración de direcciones estáticas, no se podía configurar el prefijo de las mismas. Esto no permitía seleccionar la dirección de origen correctamente.

Las reglas que se estaban utilizando eran las definidas en el manual del desarrollador de FreeBSD [FHBD], pero no estaban implementadas en completitud. Además las mismas difieren con las especificadas en el RFC 3484, Default Address Selection [RFC3484]. Por estos motivos es que se decidió reimplementar el algoritmo de selección de dirección de IP origen, a partir de las reglas definidas en el mencionado RFC, por más información ver el capitulo 2.3 del **Anexo II – Router & Nat – IP4JVM.**

A la hora de realizar una implementación de NAT para IPv6 se estudiaron distintas posibilidades, dentro de ellas, un Internet Draft, NAT66 presentado ante la Internet Engineering Task Force [IETF] “IPv6-to-IPv6 Network Address Translation”

[NAT66] y la implementación de NAT para IPv4, con las modificaciones necesarias para que funcione en IPv6.

Los **Internet Drafts** o borradores son una serie de documentos, en los que aún se está trabajando, publicados por la **Internet Engineering Task Force (IETF)** [IETF]. Estos documentos tienen una duración máxima de seis meses de validez, luego del cual deben ser actualizados a una nueva versión o serán eliminados, o pueden llegar a transformarse en un RFC.

La ilustración 3 presenta el ciclo de vida de los RFCs.

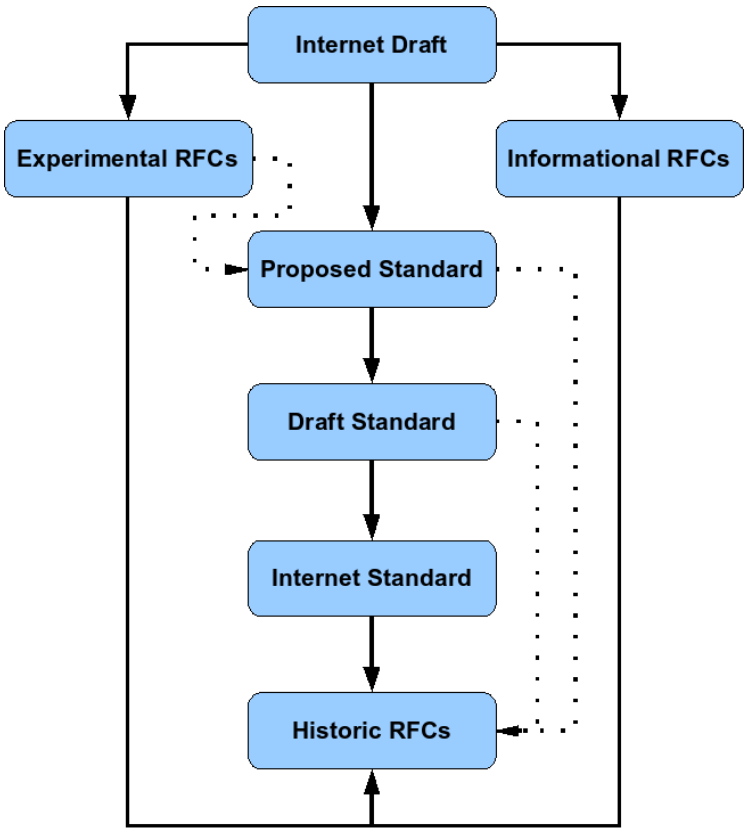


Ilustración 3: Ciclo de vida de los RFCs

Existe un estricto procedimiento que deben seguir todos los componentes de Internet para convertirse en un estándar.

Una especificación comienza como un Internet Draft el cual es un documento en el que se está trabajando y que es continuamente analizado y actualizado. No tiene un estatus oficial y tiene un ciclo de vida de 6 meses.

Luego de esta etapa el documento pasa a ser un Proposed Standard y es en este punto que la especificación se considera estable y lo suficientemente interesante para ser considerada por la comunidad de Internet. Es en este estado

que la especificación es testeada e implementada por varios grupos, si es que antes no fue implementada en el nivel de Internet Draft.

El siguiente nivel es Draft Standard, el cual es alcanzado luego de al menos 2 implementaciones independientes, interoperables y satisfactorias. Si no existen complicaciones, ni modificaciones, generalmente pasa a ser un Internet Standard.

Existen otros niveles como lo es Experimental RFCs los cuales son borradores que no necesariamente fueron desarrollados para ser un estándar de Internet, sino que son utilizados para probar situaciones experimentales. También existen los Informational RFCs que contienen información general, histórica, o tutoriales en lugar de estándares.

En el nivel Historic RFCs, se encuentran los RFCs discontinuados u obsoletos por RFCs actualizados o removidos por otras razones. El encargado de determinar el nivel de los documentos es el Internet Engineering Steering Group (IESG) [IESG] que pertenece al IETF.

Por más información sobre el ciclo de vida de los RFCs se recomienda la lectura del RFC 2026 The Internet Standards Process [RFC2026].

NAT66 podría ser implementado en un router IPv6 para mapear un prefijo de direcciones IPv6 a otro prefijo IPv6 a medida que cada paquete IPv6 transita a través del router. En su forma más sencilla, un dispositivo NAT66 se encuentra conectado a dos interfaces de red, una que es la perteneciente a la red interna, y la otra interfaz perteneciente a la red externa con conectividad a la red global de Internet. Todos los host de la red interna utilizan direcciones de un único prefijo ruteable localmente, y estas direcciones se traducirán hacia o desde direcciones de prefijos ruteables globalmente a medida que los paquetes IP transitan el dispositivo NAT66.

Como es mencionado en el borrador, en varias ocasiones, no es la intención de los autores del mismo alentar la implementación o uso del NAT66, si no que lo que se intenta es minimizar los impactos negativos en caso de que algunos implementadores decidan desarrollar un mecanismo de IPv6 NAT, y de que algunos administradores de red decidan utilizarlo. Algo similar ocurre con este proyecto, se realizó una implementación de NAT66 no con la idea de alentar o desalentar su utilización, si no que se realizó un estudio e implementación de NAT66 con la idea de proveer una herramienta que puede ser utilizada. Es mejor tenerla y optar por no utilizarla que no disponer de ella.

3.4.Dispositivos Móviles

Al momento de responder la pregunta, ¿es posible realizar un deploy del proyecto en un dispositivo móvil?, surgieron varias ramas que requerían ser

investigadas. Estas ramas fueron, que diferencias hay entre J2SE y Java Micro Edition [JME], que sistemas operativos están disponibles para los dispositivos móviles y que soporte proveen estos sistemas operativos para poder comunicarse con la interfaz de red.

Al iniciar el estudio de Java ME lo más relevante fue que la concepción de esta máquina virtual puede variar de un dispositivo móvil a otro, y no sólo por el sistema operativo que este utilice si no que también por las capacidades que este posea.

Java ME es una colección de tecnologías y especificaciones que pueden ser combinadas para construir un Java runtime especifico de forma que cumpla con los requerimientos de un dispositivo o mercado en particular.

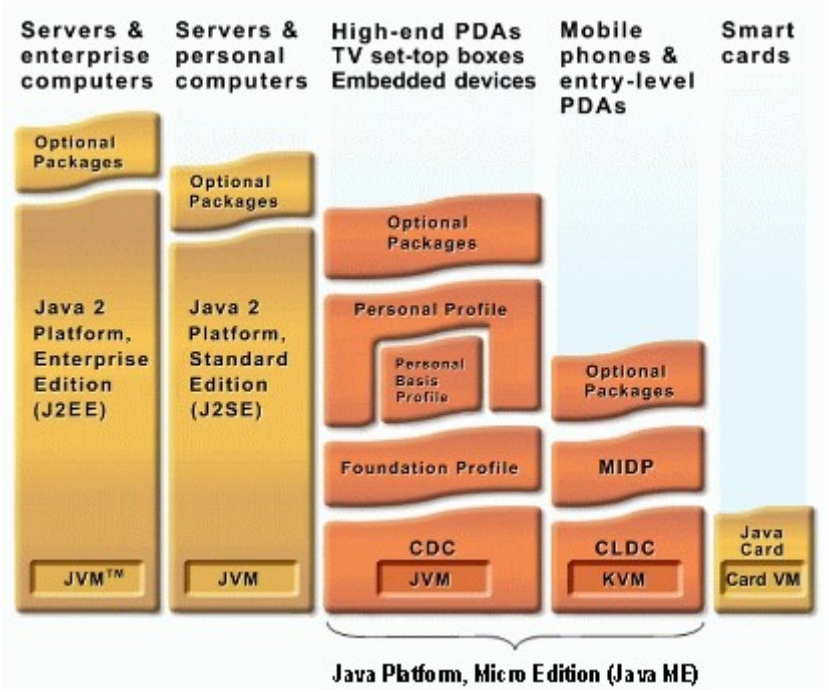


Ilustración 4: Tecnologías Java

En la ilustración 4 se puede apreciar una visión general de los componentes de la tecnología Java ME y como están relacionados con las demás tecnologías Java. Se puede ver que la tecnología Java ME esta basada en tres elementos, una configuración que provee de un conjunto básico de bibliotecas, un perfil y paquetes opcionales.

Con el tiempo la plataforma Java ME se ha dividido en dos configuraciones base, una que se ajusta a los dispositivos móviles más pequeños llamada "Connected Limited Device Configuration" (CLDC)[CLDC] y otra para dispositivos más capaces "Connected Device Configuration"(CDC)[CDC].

Cuando se paso a la etapa de estudio de los sistemas operativos se pudo apreciar que existe una amplia cantidad de estos, debido a los tiempos disponibles se decidió tomar una muestra lo más representativa posible del mercado de dispositivos móviles. Esta selección fue tomada en base a la experiencia laboral de los integrantes del proyecto y a información extraída de la web [MOBMRK] y dio como resultado los sistemas operativos Symbian OS [SOS], Android [AND], IPhone OS [IOS] y Windows Mobile [WMOB].

Esta selección de sistemas operativos no sólo abarca la gran mayoría de los dispositivos móviles disponibles en el mercado, si no que también abarca sistemas operativos que se encuentran disponibles desde los comienzos de los dispositivos móviles hasta sistemas nuevos e innovadores.

4.Cronograma y organización del trabajo

En este capítulo se presenta la asignación de tiempos y recursos en las distintas tareas.

4.1.Modalidad de trabajo

Durante el proyecto se realizaron reuniones semanales entre los dos integrantes del equipo, salvo algunas excepciones en las cuales las reuniones se dieron de forma quincenal. Los motivos de estas reuniones fueron variados y los mismos cambiaron durante el transcurso del proyecto. Durante el estudio del arte el motivo principal fue el de discernir sobre los temas estudiados con el propósito de llegar a tener una buena comprensión del todo por parte de ambos integrantes. Cuando se arribó a la etapa de análisis y diseño los motivos de las reuniones pasaban por presentar las ideas de cada integrante evaluando sus pros y contra para poder arribar a una solución para los problemas planteados. Durante la etapa de implementación de la solución las reuniones estaban motivadas en la integración del trabajo realizado por ambos integrantes durante la semana así como la designación de las tareas a realizar por estos en la semana entrante.

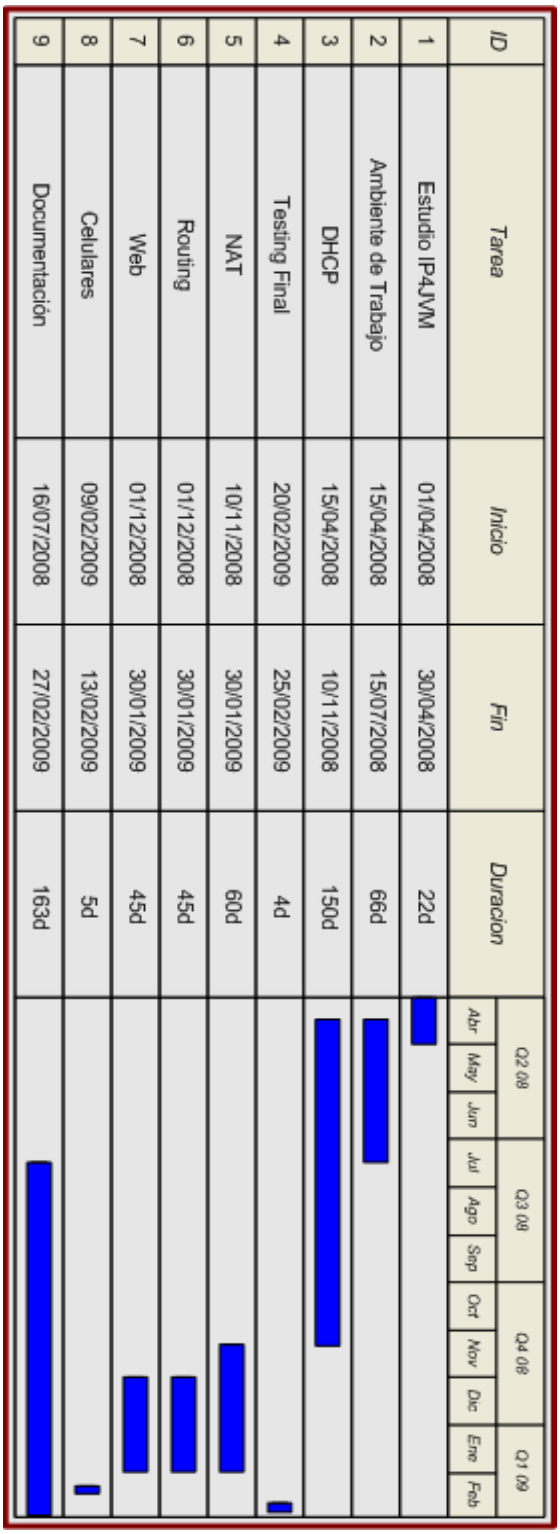
Un fuerte complemento a las reuniones mencionadas en el párrafo anterior fue la interacción por medio del chat, medio mediante el cual muchas veces se lograban disipar dudas de una forma rápida y sin necesidad de esperar a tener la reunión para charlar de la misma.

Para manejar el código se uso un sistema de control de versiones SVN para poder mantener controlados los cambios y poder colaborar de forma remota, permitiendo esto integrar cambios sin necesidad de esperar a una reunión, por más información de la configuración del SVN ver **Anexo V - Configuración del Ambiente de Testing en IP4JVM**.

Cabe destacar que las herramientas mencionadas previamente, chat y SVN, fueron de gran utilidad ya que los integrantes del equipo viven en distintas ciudades.

Durante el transcurso del proyecto también se realizaron reuniones con el tutor, estas reuniones en un principio se dieron de forma quincenal para luego pasar a ser mensuales. Ya estando bastante adelantado el proyecto las reuniones se fueron coordinando cuando se presentaba la necesidad de hacerlo.

4.2.Cronograma



ID	Tarea	Inicio	Fin	Duracion
1	Estudio IP4JVM	01/04/2008	30/04/2008	22d
2	Ambiente de Trabajo	15/04/2008	15/07/2008	66d
3	DHCP	15/04/2008	10/11/2008	150d
4	Testing Final	20/02/2009	25/02/2009	4d
5	NAT	10/11/2008	30/01/2009	60d
6	Routing	01/12/2008	30/01/2009	45d
7	Web	01/12/2008	30/01/2009	45d
8	Celulares	09/02/2009	13/02/2009	5d
9	Documentación	16/07/2008	27/02/2009	163d

Ilustración 5: Tiempo insumido

La ilustración 5 presenta el diagrama de Gantt de la tareas, con los tiempos insumidos por estás, que fueron realizadas con el motivo de lograr cumplir los objetivos planteados en el capítulo 2 de este documento. Los tiempos para la implementación de DHCPv6 y de Router & NAT, se pueden ver con mayor grado de detalle en los documentos **Anexo I – DHCPv6 – IP4JVM** y **Anexo II – Router & NAT – IP4JVM** en el capítulo 4.

A continuación se presenta la tabla de tiempos estimados y tiempos insumidos para la realización de cada tarea.

Tarea	Tiempo Estimado	Tiempo Insumido
Estudio IP4JVM	3 semanas	**6 semanas**
Configuración ambiente de trabajo	6 semanas	**9,5 semanas**
DHCPv6	18 semanas	**21,5 semanas**
Implementación Router & NAT	11 semanas	**13 semanas**
Aplicación web	6 semanas	**6,5 semanas**
Dispositivos móviles	1 semana	1 semana
Testing Final	1 semana	1 semana
Documentación	20 semanas	**23 semanas**
Total	36 semanas	**48 semanas**

Ilustración 6: Tabla de tiempos estimados e insumidos

Si bien en las ilustraciones 5 (Tiempo insumido) y 6 (Tabla de tiempos estimados e insumidos) no se aprecian períodos de inactividad los mismos existieron pero fueron considerados dentro de los tiempos estimados y contabilizados en los tiempos insumidos.

Como se puede apreciar en la tabla de tiempos estimados e insumidos, casi todas las tareas, a excepción de las tareas de dispositivos móviles y de testing final, sufrieron desviaciones importantes con respecto a lo inicialmente planificado. Las mayores desviaciones se dieron al comienzo del proyecto, esto se debió a que la curva de aprendizaje fue más lenta de lo que se consideró al comenzar el mismo. El estudio del stack llevo el doble del tiempo considerado y si bien fue realizado en paralelo con la configuración del ambiente de trabajo, que llevó más de una vez y media el tiempo que se consideró inicialmente, no se pudo cumplir con la planificación inicial. Luego a medida que pasaron las etapas, se tenía un

conocimiento más profundo del proyecto IP4JVM y el estudio del mismo se concentró en lo que se buscaba específicamente lo que permitió que los tiempos de estudio se redujeran en las siguientes etapas. Como ya se mencionó previamente, en los documentos anexos **Anexo I – DHCPv6 – IP4JVM** y **Anexo II – Router & NAT – IP4JVM**, se encuentra un estudio más detallado de los tiempos estimados e insumidos. Para la aplicación web la diferencia de tiempos fue de apenas media semana. Mientras que para la etapa de dispositivos móviles y el testing final no existieron diferencias en lo planificado y el tiempo que insumieron los mismos. Por último la desviación de la documentación fue de 3 semanas de lo proyectado inicialmente.

En total el trabajo se realizó en 48 semanas, culminando en el mes de marzo de 2009, a diferencia de las 36 semanas planificadas inicialmente para culminar en diciembre de 2008. El total estimado de semanas efectivas trabajadas, no se realizó un control de horas riguroso, se encuentra en el entorno de 40 semanas.

5.Resultados y conclusiones

Durante el transcurso del proyecto se fueron cumpliendo los objetivo de forma gradual, se realizaron estudios del arte sobre los temas primordiales presentes en los mismos, se evaluaron distintas alternativas para poder cumplir con estos arribando finalmente a las soluciones presentadas en este documento y complementadas en los documentos anexos a este.

La primera etapa del proyecto consistió en el planteo de incorporar al proyecto IP4JVM la funcionalidad de configuración de direcciones mediante el protocolo DHCPv6.

Luego de considerar cumplido el primer objetivo planteado se pasó a afrontar los siguientes objetivos, la implementación de las funcionalidades de router para el nodo IP4JVM, una solución NAT para IPv6, una aplicación web para permitir configurar el proyecto, y el estudio sobre la posibilidad de implantación del stack en dispositivos móviles.

En los siguientes puntos se presentan los logros, limitaciones y mejoras de los componentes diseñados e implementados, así como las conclusiones finales del proyecto.

5.1.DHCPv6

El nivel del producto obtenido es comprable con el nivel de productos que se encuentran en producción, una prueba de esto es el nivel de éxito obtenido durante los tests realizados al proyecto mediante el proyecto TAHI, en comparación a los resultados obtenidos por aplicaciones que se encuentran en producción, como por ejemplo el Dibbler[DIBER], por más información ver la sección 4.2.24 del **Anexo VI – Testing en IP4JVM**.

Además se logro una solución que permite ser utilizada fácilmente en otras implementaciones del stack IPv6, debido a que si bien se encuentra vinculado al proyecto IP4JVM, la solución implementada se basó en un diseño de un componente que permite la comunicación con el stack IP4JVM por medio de interfaces de sistema. Para que otro stack IPv6 utilice este componente sólo debe implementar las interfaces definidas.

Si bien la configuración DHCPv6 se puede realizar en todas sus formas, no todos los parámetros que pueden recibirse o solicitarse están soportados. Un ejemplo de esto es el manejo de servidores DNS, NTP, y de dominio. Vale la pena destacar que en caso de recibir una de estás opciones u otras el sistema simplemente las descarta, pero considera el resto de las opciones que reconoce.

Por más información ver capítulo 5, Resultados y conclusiones **Anexo I – DHCPv6 – IP4JVM**.

5.2.Router & NAT

Se implementaron las funcionalidades mínimas de un router IPv6, la única funcionalidad extra que se implementó fue el ICMPv6Redirect. Para esto se debió modificar el algoritmo de selección de dirección de origen existente el cual se encontraba implementado, y no de forma completa, siguiendo las reglas definidas en el manual de desarrollador de FreeBSD [FHBD], por las definidas en el RFC 3484 Default Address Selection for Internet Protocol version 6 (IPv6) [RFC3484].

Por el lado de IPv6 a IPv6 NAT, primero se consideraron las distintas alternativas posibles, resolviendo luego la implementación del Internet Draft NAT66 IPv6-to-IPv6 Network Address Translation. Al implementar un borrador se tomó el riesgo de implementar una especificación que puede no llegar a ser un estándar, o si llegar a serlo pero sufriendo modificaciones en el proceso de estandarización. Luego de la culminación de la implementación y de la generación de la documentación de la etapa de Router & NAT66, el día 9 de marzo, en la lista de NAT66 [LSTNAT] se publicó una actualización del Internet Draft en su tercera versión, el cual aún no fue publicado en la IETF por problemas técnicos.

Por más información ver capítulo 5, Resultados y conclusiones en el **Anexo II – Router & NAT – IP4JVM**.

5.3.Aplicación web

Se obtuvo una aplicación web amigable que permite realizar configuraciones y test sobre el proyecto, la cual a su vez corre sobre el mismo.

Si bien la aplicación es totalmente independiente si el sistema tiene o no soporte nativo para IPv6, no todas las aplicaciones Java funcionarán correctamente sobre el mismo sin realizar modificaciones. Esto se debe a que algunas de ellas, por ejemplo el Tomcat, invocan operaciones relacionadas al sistema operativo para saber si el mismo brinda o no un soporte de IPv6.

Por más información ver capítulos 3 y 4 en el **Anexo III – Aplicación Web – IP4JVM**.

5.4.Dispositivos móviles

Esta etapa comenzo con una simple pregunta, ¿es posible realizar el deploy del proyecto en un dispositivo móvil?

La respuesta más apropiada es "quizas", esto se debe a que las preguntas que se deben de responder para poder negar o afirmar que es posible el deploy del proyecto son muchas y no todas tienen una respuesta clara.

Por un lado tenemos las distintas configuraciones que puede tener una máquina virtual Java es su versión microedition, en este caso se llegó a dilucidar que era necesaria una configuración CDC o superior para que el proyecto pudiera correr, descartandose la configuración CLDC.

Por otro lado se encuentran los sistemas operativos que están disponibles en los distintos dispositivos móviles. Como se mencionó anteriormente estos son muchos con lo cual se tomó una muestra representativa de los más difundidos en el mercado. De los sistemas operativos nos interesaban conocer dos datos primordiales, si podían correr la máquina virtual Java Microedition en su versión CDC o superior y si era posible tener acceso a los paquetes de red manejados por el dispositivo móvil.

Como respuestas a estas incógnitas se concluyó que en ninguno de los sistemas operativos se podía afirmar que era posible realizar la instalación de la configuración CDC de la máquina virtual Java ME. Pero si se puede afirmar que con una recompilación del código fuente de Windows Mobile se puede dar soporte para la mencionada configuración de Java ME. Para IPhone OS es posible tecnológicamente pero no legalmente actualmente. Por último se tiene el sistema operativo Android, el cual de por si es una máquina virtual Java corriendo sobre un kernel de GNU/Linux. Este sistema no presenta impedimentos técnicos con respecto a Java, siempre y cuando sea posible realizar las modificaciones necesarias para poder adaptar su máquina virtual al igual que se realizó con la máquina virtual OpenJDK.

Por último resta responder si es posible acceder a los paquetes de red del dispositivo móvil. En este caso la respuesta es sencilla para IPhone OS, Windows Mobile y Android, ya que los 3 sistemas operativos pueden correr la biblioteca libpcap en sus distintas versiones para cada sistema operativo.

Por más información ver capítulo 3, Conclusiones en el **Anexo IV – Dispositivos móviles – IP4JVM**.

5.5.Logros generales

Se comenzó el trabajo en la revisión 529 del SVN del proyecto IP4JVM, la cual contaba con único subproyecto, con 16.666 líneas de código, 259 clases y 4 interfaces.

En el momento actual el proyecto consta de 4 subproyectos, el subproyecto inicial IP4JVM, el subproyecto DHCPv6, el subproyecto Common, y el subproyecto web. Cada uno de ellos cuenta con 19.191, 3.351, 26 y 1.319 líneas de código respectivamente, sumando en total 23.987 líneas de código.

En materia de clases el subproyecto IP4JVM consta de 273 clases y 4 interfaces, el subproyecto DHCPv6 tiene 56 clases, el subproyecto web 2 y el subproyecto Common 1 clase y 2 interfaces. En total el proyecto actualmente consta de 332 clases y 6 interfaces.

A excepción de las líneas de código JSP el resto de las mediciones fueron realizadas mediante metrics[MET], en su versión 1.3.6.

El último commit de código se realizó en la revisión 734.

5.6.Trabajo futuro y posibles mejoras

Este proyecto aún dista mucho de estar culminado, los trabajos y mejoras que se pueden realizar sobre el mismo son varios, algunos de los más interesantes son:

- Soporte para el todos los parámetros e información recibida en una configuración DHCP como puede ser los parámetros referidos a la configuración de DNS.
- Soporte de stack de protocolos 802.11 en capa de enlace.
- Implementación de protocolos de Routing como por ejemplo Distancia Vector, IGRP, OSFP, EIP y BGP.
- IPsec
- Poder realizar manejo de protocolos de seguridad.
- Poder agregar extensiones, como puede ser un firewall (reglas para recepción de paquetes, IPs, puertos, etc).
- Realización de un paquete de configuración/instalación.
- Generación de una página web del proyecto.

5.7.Resumen

Como ya se mencionó los objetivos del proyecto se consideran cumplidos aunque aún se pueda profundizar más sobre los temas tratados durante el transcurso del mismo.

En la presente iteración los avances en conectividad han sido mejorados, se agregaron protocolos nuevos, se mejoraron los protocolos existentes y se agregaron funcionalidades nuevas que permiten configurar el nodo como un router. Esto último también permitió implementar una nueva herramienta que permite configurar un nodo como un dispositivo NAT66 para realizar la interconexión de redes.

También se mejoró la usabilidad del proyecto mediante una aplicación web que permite configurar los distintos parámetros del sistema.

Otro gran avance fue el de poder comprobar la correcta interoperabilidad con otros dispositivos y conformidad de los RFCs mediante los tests del proyecto TAHI.

Además se presentó un estudio sobre las modificaciones y las herramientas a utilizar para poder realizar el deploy del stack sobre dispositivos móviles.

5.8.Conclusiones

Si bien se comparan algunos de los resultados obtenidos con aplicaciones en producción el proyecto está muy lejos de competir contra lo que es un soporte nativo de IPv6 por parte de un sistema operativo. Esto se puede apreciar en los tiempos de transmisión de paquetes de tamaños considerables como los realizados en el **Anexo VI – Testing en IP4JVM – IP4JVM** en el capítulo 5.4 Benchmarking.

Pero si puede resultar en una herramienta muy útil para ser utilizada de forma didáctica con algunas modificaciones (por ejemplo el instalador), ya que permite incorporar de forma sencilla algunos protocolos. De esta forma los estudiantes podrían lidiar con los distintos protocolos de redes sin tener que preocuparse del manejo de memoria y punteros que implicaría tener si se realizara la misma tarea en lenguajes como C o C++.

IP4JVM es una herramienta de gran utildad para realizar implementaciones de nuevos protocolos o modificaciones de ya existentes en una forma sencilla. Una prueba de esto es la implementación que se realizó durante este proyecto del Internet Draft NAT66.

Se puede afirmar luego de culminada esta tercera iteración sobre el proyecto este cuenta con una arquitectura estable y extensible.

En conclusión IP4JVM es una herrmienta que cuenta con cierto grado de madurez pero que aún dista mucho de poder ser utilizada en un ambiente de producción pero si puede llegar a contar con un gran futuro en ambientes académicos y de investigación.

6.Referencias

Las referencias presentadas en este punto corresponden a las referencias de todo el proyecto.

[AND]

•Página del Sistema operativo Android - http://code.google.com/intl/es-ES/android/

[AWT]

•Abstract Windows Toolkit - http://java.sun.com/products/jdk/awt/

[BREW]

•Sistma operativo BREW - http://brew.qualcomm.com

[BSD]

•Berkeley Software Distribution – http://www.bsd.org/

[CDC]

•Connected Device Configuration- http://java.sun.com/products/cdc/

[CLDC]

•Connected Limited Device Configuration- http://java.sun.com/products/cldc/

[DIAN]

•Sistema operativo Debian - http://www.debian.org/

[DIBER]

•Dibbler, Implementación de DHCPv6 portable - http://klub.com.pl/

[DVIK]

•Máquina virtual Dalvik - http://www.dalvikvm.com/

[EPL]

•Eclipse Public License - http://www.eclipse.org/legal/epl-v10.html

[EPSE]

•IDE para desarrollo Java - http://www.eclipse.org/

[ETHII]

- Ethernet II - http://www.yale.edu/pclt/COMM/ETHER.HTM

[FBSD]

- Free BSD - http://www.freebsd.org/

[FHBD]

- FreeBSD Developers Handbook - http://www.freebsd.org/doc/en/books/developers-handbook/ipv6.html

[GWT]

- Google Web Toolkit - http://code.google.com/intl/es-ES/webtoolkit/

[HIPV6]

- Taller de IPv6 realizado por el UY6TF - http://www.uy6tf.org.uy/?q=haciaipv6 g

[IETF]

- The Internet Engineering Task Force - http://www.ietf.org/rfc.html

[INRIA]

- l'Institut National de Recherche en Informatique et en Automatique - http://www.inria.fr/index.en.html

[IOS]

- Página del sistema operativo IPhone OS
  - http://developer.apple.com/iphone
  - http://developer.apple.com/iphone/gettingstarted/docs/iphoneosoverview.action

[IP4JVM]

- Documentación proyecto previo - http://www.fing.edu.uy/~asabigue/prgrado/2007IP4JVM_Abelenda_Corrales.pdf

[JME]

- Java Microedition - http://java.sun.com/javame/index.jsp

[JNI]

- API de Java que permite vincular codigo Java con bibliotecas no implementadas en Java - http://java.sun.com/javase/6/docs/technotes/guides/jni/

[JVM]

- Máquina Virtual Java - http://java.sun.com/javase/6/docs/technotes/guides/vm/index.html

[JVPOS]

- Blog de Sun - http://blogs.sun.com/ontherecord/entry/sun_announces_intent_to_create

[KVM]

- The K virtual machine - http://java.sun.com/products/cldc/wp/

[MET]

- Metrics – http://metrics.sourceforge.net/

[MMRK]

- Información comercial sobre dispositivos móviles
  - http://marketshare.hitslink.com/mobile-phones.aspx?qprid=55
  - http://www.techcrunch.com/2008/09/03/can-the-iphone-beat-symbian-os/
  - http://www.canalys.com/pr/2008/r2008112.htm

[MSDN]

- Página de información tecnica del sistema operativo Windows Mobile -
  - http://msdn.microsoft.com/en-us/windowsmobile/default.aspx
  - http://www.microsoft.com/spain/windowsmobile/default.mspx

[NAT66]

- Internet Draft
  - IPv6-to-IPv6 Network Address Translation (NAT66)
  - http://tools.ietf.org/html/draft-mrw-behave-nat66-01
  - 3 de Noviembre de 2008
  - M. Wasserman, F. Baker

[OPESU]

- Open Suse – http://opensuse.org/

[OPJDK]

- The Open-Source JDK Community - http://openjdk.java.net/

[PALOS]

- Sistma operativo Palm OS - http://www.palm.com/

[PCAP]

- Bibilioteca que permite capturar y enviar trafico de paquetes de red
  - http://www.tcpdump.org/
  - http://www.winpcap.org/
  - http://libpcappp.sourceforge.net/

[REGO]

- Ipv6 Ready Logo - http://www.ipv6ready.org/

[RFC]

- RFC 768
  - User Datagram Protocol
  - http://www.ietf.org/rfc/rfc768.txt
  - Agosto 1980
  - G. Postel
- RFC 793
  - Transmition Control Protocol
  - http://www.ietf.org/rfc/rfc793.txt
  - Septiembre 1981
  - J. Postel
- RFC 951
  - Bootstrap Protocol (BOOTP)

- http://www.ietf.org/rfc/rfc951.txt
- Septiembre 1985
- Bill Croft, Jhon Gilmore

- RFC 1071
  - Computing the Internet Checksum.
  - http://www.ietf.org/rfc/rfc1071.txt
  - Septiembre 1988
  - R. Braden, D. Borman, C. Partridge
- RFC 1981
  - Path MTU Discovery for IP version 6
  - http://www.ietf.org/rfc/rfc1981.txt
  - Agosto 1996
  - J. McCann, S. Deering, J. Mogul
- RFC 2131
  - Dynamic Host Configuration Protocol
  - http://www.ietf.org/rfc/rfc2131.txt
  - Marzo 1997
  - R. Droms
- RFC 2460
  - Internet protocol, Version 6 (IPv6) Specification
  - http://www.ietf.org/rfc/rfc2460.txt
  - Diciembre 1998
  - S. Deering, R. Hinden
- RFC 2663
  - IP Network Address Translator (NAT) Terminology and Considerations
  - http://www.ietf.org/rfc/rfc2663.txt

- Septiembre 1988
- P. Srisuresh, Lucent Technologies

- RFC 2993
  - Architectural Implications of NAT
  - http://www.ietf.org/rfc/rfc2993.txt
  - Noviembre 2000
  - T. Haing
- RFC 3022
  - Traditional IP Network Address Translator (Traditional NAT)
  - http://www.ietf.org/rfc/rfc3022.txt
  - Enero 2001
  - P. Srisuresh, K. Egevang
- RFC 3315
  - Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
  - http://www.ietf.org/rfc/rfc3315.txt
  - Julio 2003
  - R. Droms, J. Bound, B. Volz, T. Lemon, C. Perkins, M. Carney
- RFC 3319
  - Dyanamic Host Configuration Protocol (DHCPv6) Options for Session Initiation Protocol (SIP) Servers
  - http://www.ietf.org/rfc/rfc3319.txt
  - Julio 2003
  - H. Schulzrinne, B. Volz
- RFC 3633
  - IPv6 Prefix Options for Dynamic Host Configuration Protocol (DHCP) version 6
  - http://www.ietf.org/rfc/rfc3633.txt

- Diciembre 2003
- O. Troan, R. Droms

- RFC 3646
  - DNS Configuration options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6)
  - http://www.ietf.org/rfc/rfc3646.txt
  - Diciembre 2003
  - R. Droms

- RFC 3736
  - Stateless Dynamic Host Configuration Protocol (DHCP) Service for IPv6
  - http://www.ietf.org/rfc/rfc3736.txt
  - Abril 2004
  - R. Droms

- RFC 3844
  - IETF Problem Resolution Process
  - http://www.ietf.org/rfc/rfc3844.txt
  - Agosto 2004
  - E. Davies, J. Hofmann

- RFC 3849
  - IPv6 Address Prefix Reserved for Documentation
  - http://www.ietf.org/rfc/rfc3849.txt
  - Julio 2004
  - G. Huston, A. Lord, P. Smith

- RFC 3879
  - Deprecating Site Local Addresses
  - http://www.ietf.org/rfc/rfc3879.txt
  - Septiembre 2004

- C. Huitema, B. Carpenter

- RFC 3898

  - Network Information Service (NIS) Configuration Options for Dynamic Host Configuration Protocol for IPv6 (DHCPv6).

  - http://www.ietf.org/rfc/rfc3898.txt

  - Octubre 2004

  - V. Kalusivalingam

- RFC 4193

  - Unique Local IPv6 Unicast Addresses

  - http://www.ietf.org/rfc/rfc4193.txt

  - Octubre 2005

  - R. Hinden, B. Haberman

- RFC 4291

  - IP Version 6 Addressing Architecture

  - http://www.ietf.org/rfc/rfc4291.txt

  - Febrero 2006

  - R. Hinden,S. Deering

- RFC 4443

  - Internet Control Message Protocol (ICMPv6) for the Internet Protocol Version 6 (IPv6) Specification

  - http://www.ietf.org/rfc/rfc4443.txt

  - Marzo 2006

  - A. Canta, S. Deering, M. Gupta

- RFC 4861

  - Neighbor Discovery for IP version 6 (IPv6)

  - Septiembre 2007

  - http://www.ietf.org/rfc/rfc4861.txt

- T. Narten, E. Nordmark, W. Simpson, H. Soliman

- RFC 4864
  - Local Network Protection for IPv6
  - http://www.ietf.org/rfc/rfc4864.txt
  - Mayo 2007
  - G. Van de Velde, T. Hain, R. Droms, B. Carpenter, E. Klein
- RFC 4862
  - IPv6 Stateless Address Auto configuration
  - http://www.ietf.org/rfc/rfc4862.txt
  - Septiembre 2007
  - S. Thomson, T. Narten,T. Jinmei

[RIM]

- Sistma Operativo RIM - http://www.rim.com/

[RVD]

- radvd, Linux IPv6 Router Advertisement Daemon - http://www.litech.org/radvd/

[SAJE]

- Sistma operativo Savaje -http://www.sun.com/software/javafx/mobile/

[SCPE]

- Plugin para el IDE eclipse que provee soporte para el manejo de versiones - http://subclipse.tigris.org/

[SIXXS]

- Página para el registro de ULAs - http://www.sixxs.net/

[SOS]

- Sistema operativo Symbian OS - http://www.symbian.com/index.asp

[SUN]

- Sun Microsystems Homepage - http://www.sun.com/

[TAHI]

- Tahi Project, test de conformidad e interoperabilidad para IPv6 - http://www.ietf.org/index.html

[TCAT]

- Implementación de las tecnologías Java Servlet y JavaServer Pages - http://tomcat.apache.org/

[UBU]

- Sistema operativo Ubuntu - http://www.ubuntu-es.org/

[UY6TF]

- IPv6 Task Force Uruguayo - http://www.uy6tf.org.uy/

[VARE]

- VMware, es un sistema de vitalización por software - http://www.vmware.com/

[VIRB]

- Virtual Box – http://www.virtualbox.org

[WARK]

- Wireshark, analizador de tráfico de redes - http://www.wireshark.org/

[WINXP]

- Sistema operativo Windows Xp - http://www.microsoft.com/spain/windowsxp/pro/evaluation/sysreqs.mspx

[WMOB]

- Sistema operativo Windows Mobile - http://www.microsoft.com/windowsmobile/en-us/default.mspx

[WURLF]

- Archivo XML que contiene información sobre las caracteristicas de los distintos dispositivos móviles- http://wurfl.sourceforge.net/

[WVST]

- Sistema operativo windows vista - http://www.microsoft.com/latam/windows/windows-vista/

7.Índice de ilustraciones