VirtShell
Framework para aprovisionamiento de soluciones virtuales

Carlos Alberto Llano R.
Escuela de Ingeniería de Sistemas y Computación
Universidad del Valle
Cali, Valle del Cauca
Email: carlos_llano@hotmail.com

John Alexander Sanabria
Escuela de Ingeniería de Sistemas y Computación
Universidad del Valle
Cali, Valle del Cauca
Email: john.sanabria@correounivalle.edu.co

***Resumen*—The abstract goes here.**

I. Introducción

La aparición de ambientes de computación centrados en la nube, los cuales se caracterizan por ofrecer servicios bajo demanda, ha favorecido el desarrollo de diversas herramientas que apoyan los procesos de aprovisionamiento en demanda de servicios y ambientes de computación orientados al procesamiento de tareas de larga duración y manejo de grandes volúmenes de datos. Estos ambientes dinámicos de computación son desarrollados mayormente a través de técnicas de programación ágil las cuales se caracterizan por ofrecer rápidos resultados e integración a gran escala de componentes de software. Es así como los equipos de DevOps [1] se convierten en un elemento fundamental ya que potencia la estabilidad y uniformidad de los distintos ambientes de prueba y producción de modo que los procesos de integración y despliegue se hagan de forma automatizada.

Las herramientas de aprovisionamiento automático de infraestructura son el eje central de estos equipos ya que es a través de ellas que el personal de desarrollo y operaciones son capaces de hablar un mismo lenguaje y establecer los requerimientos y necesidades a satisfacer. Sin embargo, las herramientas actuales de aprovisionamiento adolecen de servicios que faciliten la especificación de infraestructura a través de un API [2] estandarizado que posibilite la orquestación del despliegue de infraestructura a través de Internet.

En este artículo se presenta una herramienta de aprovisionamiento con orientación a servicios que permite el despliegue y orquestación de plataformas y servicios a través de un API RESTful [3]. Además de lo mencionado anteriormente, la artículo consta de 8 secciones.

La segunda sección presenta el planteamiento del problema, en donde se aclara el objeto y alcance del trabajo realizado. La definición del marco teórico que permite entender la importancia de la virtualización en la actualidad, las técnicas de virtualización usadas y la sinopsis de las soluciones de aprovisionamiento mas conocidas actualmente, son presentadas en la tercera sección.

En la cuarta sección se introduce y elabora la arquitectura planteada en VirtShell. Se describe los requisitos que se tuvieron en cuenta para elaborar la estructura del framework, las alternativas estudiadas y se reseña los módulos y sus características que conforman a VirtShell.

Las siguientes 4 secciones se encargan de describir cada uno de los módulos diseñados, ilustrando sus funcionalidades y la forma en que interactúan de manera conjunta para administrar la infraestructura y realizar el aprovisionamiento de los recursos virtualizados. Adicionalmente se muestran ejemplos del uso del API.

En la ultima sección se muestra la documentación del API de VirtShell. Se indican los recursos y los métodos HTTP con que cuenta cada módulo que permiten interactuar con el API.

II. Problema

En la actualidad, con la creciente adopción de modelos de computación como el *Cloud Computing* [4] y el *Grid*

[1] DevOps consiste en traer las prácticas del desarrollo ágil a la administración de sistema y el trabajo en conjunto entre desarrolladores y administradores de sistemas. DevOps no es una descripción de cargo o el uso de herramientas, sino un método de trabajo enfocado a resultados.

[2] API: Application Programming Interface, conjunto de subrutinas, funciones y procedimientos que ofrece un software para ser utilizado por otro software como una capa de abstracción.

[3] RESTful hace referencia a un servicio web que implementa la arquitectura REST

[4] conocida también como servicios en la nube, es un paradigma que permite ofrecer servicios de computación a través de una red, que usualmente es Internet.

Computing [5], los ambientes computacionales se han tornado cada vez más sofisticados y complejos, requiriendo de soluciones que traten de manera integral el aprovisionamiento de diferentes servicios sobre ambientes virtuales capaces de atender la variable demanda computacional, a través del despliegue de infraestructuras elásticas de computación.

Hoy en día, se encuentran diversas soluciones que abordan el problema de aprovisionamiento usando diferentes enfoques para el despliegue y orquestación de plataformas y servicios. Sin embargo, los enfoques actuales carecen de mecanismos de comunicación que permitan interoperar entre diferentes aplicaciones; En general, las soluciones actuales presentan dificultades para ser accedidas en una red, como internet y ejecutadas de manera remota.

En consecuencia, para lograr la interoperabilidad con los actuales modelos de computación, este proyecto propone como objetivo principal, plantear el diseño de un framework orientado al web, cuyas partes soporten una arquitectura que permita el eficiente aprovisionamiento de software de manera automática para ambientes virtualizados. De igual modo, se propone realizar la ejemplificación del framework en un ambiente virtualizado.

Para lograrlo será necesario evaluar diferentes estilos arquitectónicos, realizar un modelo del framework, definir la plataforma en la que se realizará la ejemplificación y definir el mecanismo de aprovisionamiento que se utilizará para aprovisionar ambientes en máquinas virtuales.

III. Estado del arte

La tecnología de virtualización y la computación en la nube son dos áreas de investigación muy activas. La investigación que produce estos campos es demasiada para enumerar. Cada una tiene múltiples conferencias de investigación. Por ejemplo, la ACM Symposium on Cloud Computing (SOCC) [2] es uno de los lugares más conocidos en investigación de la computación en la nube y tecnologías de virtualización.

La importancia de estas dos areas en la industria radica en que las nubes han transformado la forma de hacer computación. En general una empresa que use los servicios de una nube no necesita preocuparse tanto acerca de la administración de la infraestructura, las copias de seguridad, el mantenimiento, la depreciación, fiabilidad, rendimiento y tal vez de la seguridad. Estas tareas son ahora realizadas por los proveedores de la nube en otro lugar fuera de la empresa que los contrata.

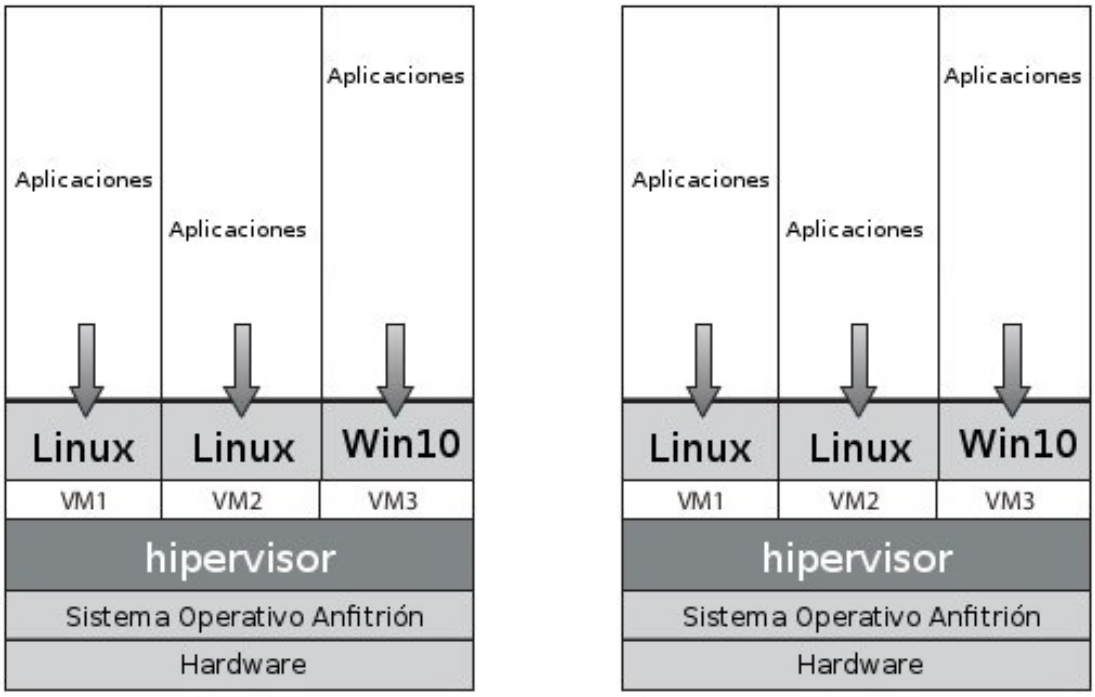
En la actualidad existen una gran oferta de nubes. Algunas de estas son públicas y están disponibles para cualquiera que esté dispuesto a pagar por el uso de los recursos, las demás son privadas para una organización. Del mismo modo, diferentes nubes ofrecen cosas diferentes. Algunas dan a sus usuarios el acceso a hardware físico, pero la mayoría permiten virtualizar sus entornos. Algunas ofrecen las máquinas virtuales, desnudas o no, y nada más, pero otras ofrecen software que está listo para utilizar y se pueden combinar de manera interesante, o plataformas que hacen que sea fácil para sus usuarios desarrollar nuevos servicios [3].

En ese contexto, una de las principales características de la computación en la nube es la virtualización, la cual crea la ilusión de múltiples máquinas (virtuales), cada una potencialmente ejecuta un sistema operativo completamente, diferente usando el mismo hardware físico. La virtualización se puede aplicar a computadoras, sistemas operativos, dispositivos de almacenamiento de información, aplicaciones o redes. Esto permite que las empresas ejecuten mas de un sistema virtual, ademas de múltiples sistemas operativos y aplicaciones, en un único servidor, de esta manera se logra economía de escala y una mayor eficiencia.

III-A. Técnicas de Virtualización

Actualmente predominan dos técnicas de virtualización. La primera técnica se denomina virtualización de hardware y consiste en que el software subyacente que ejecuta las máquinas virtuales conocido como hipervisor, crea y corre maquinas virtuales proporcionando una interfaz que es idéntica a la del servidor físico (también conocido como maquina anfitriona). El hipervisor, interactúa directamente con la CPU en el servidor físico, ofreciendo a cada uno de los servidores virtuales una total autonomía e independencia (Figura **??**). Incluso pueden coexistir en una misma maquina distintos servidores virtuales funcionando con distintos sistemas operativos. Esta técnica es la mas desarrollada y hay diferentes productos que cada fabricante ha ido desarrollando y adaptando, como por ejemplo Xen, KVM, VMWare y VirtualBox.

[5]Un grid es un sistema de computación distribuido que permite coordinar computadoras de diferente hardware y software y cuyo fin es procesar una tarea que demanda una gran cantidad de recursos y poder de procesamiento.



(a) Case I

(b) Case II

Figura 1: Simulation results for the network.

La segunda técnica es conocida como virtualización del sistema operativo. En esta técnica lo que se virtualiza es el sistema operativo completo el cual corre directamente sobre la maquina física. A este tipo de máquinas virtuales se les denomina contenedores, los cuales acceden por igual a todos los recursos del sistema (Figura 2). Esta técnica tiene una ventaja que es a su vez una desventaja: todas las máquinas virtuales usan el mismo Kernel [6]que el sistema operativo, lo que reduce los errores y multiplica el rendimiento, pero a su vez solo puede haber un mismo tipo de sistema operativo en los contenedores, no se puede combinar Windows, Linux, Etc. Este sistema también es un acercamiento a lo que seria una virtualización nativa [7].

De hecho, sin importar la técnica de virtualización que se use, la instalación de una maquina virtual (o de un contenedor) requiere normalmente de la generación e instalación de una imagen y a su vez de la instalación y configuración de paquetes de software. Estas tareas generalmente son realizadas por los técnicos de los proveedores de la nube. Cuando un usuario de la nube solicita un nuevo servicio o mas capacidad de computo, el administrador selecciona la imagen apropiada para clonar e instalar en los nodos de la nube. Si no existe una imagen apropiada para los requerimientos del cliente, se crea y configura una nueva que cumpla con la solicitud. Esta creación de una nueva imagen puede ser realizada modificando la imagen mas cercana de las ya existentes. En el momento de la creación optima de la imagen un administrador puede tener dificultades y preguntas como: ¿cuál es la mejor configuración?, ¿cuáles paquetes y sus dependencias deberían ser instaladas? y ¿cómo encontrar una imagen que mejor llene las expectativas?. Esto hace que la automatización y simplificación de este proceso sea una prioridad para los proveedores, ya que la depedencia entre entre paquetes de software y la dificultad de mantenimiento agrega tiempo a la creación de las máquinas virtuales. En otras palabras, uno de los principales objetivos de los proveedores de la nube es brindar mas flexibilidad y agilidad a la hora de satisfacer los requerimientos de los usuarios finales.

[6]Kernel es un software que constituye una parte fundamental del sistema operativo, y se define como la parte que se ejecuta en modo privilegiado (conocido también como modo núcleo)

[7]Tipo de virtualización en que intervienen las características del hardware. Los fabricantes preparan, sobre todo, los procesadores para que maquinas virtuales puedan trabajar con ellos más directamente.

Figura 2: Contenedores

III-B. Soluciones de aprovisionamiento

En el mercado existen muchas soluciones que permiten la interacción con diferentes ambientes de virtualización. Estas soluciones usan diferentes enfoques para realizar despliegues de software en las máquinas virtuales de manera rápida, controlada y automática. Sin embargo la mayoría de las soluciones no tienen la capacidad de manejar de manera simultanea las dos técnicas de virtualización antes mencionadas, algunas se centran solo en manejar máquinas virtuales y otras pocas solo hacen aprovisionamiento sobre contenedores.

Así mismo, hay soluciones de aprovisionamiento que han incorporado su propio lenguaje buscando mayor flexibilidad y fácil configuración de las tareas. Sin embargo, esto implica incorporar una curva de aprendizaje bastante alta, lo que se traduce en un gran esfuerzo inicial para contar con toda la infraestructura automatizada. En ese mismo orden de ideas, existen a su vez, soluciones cuya curva de aprendizaje es mucho menor lo que las hace mas atractivas para muchos ingenieros de Tecnologías de la Información (TI).

Adicionalmente, así como se encuentran soluciones o herramientas de aprovisionamiento de desarrollo propietario que cobran por sus funcionalidades mas importantes o por el numero de máquinas que pueden aprovisionar, existen herramientas de código abierto o de uso libre que permiten trabajar con un número considerable de máquinas virtuales. Al revisar alrededor de 40 diferentes herramientas de aprovisionamiento se logro identificar dos características que no se encuentran en las soluciones actuales. La primera trata de la ausencia de una interfaz o API web para realizar aprovisionamiento remoto y la segunda se refiere a que las herramientas se limitan a aprovisionar las máquinas virtuales sin ofrecer mecanismos de administración y monitoreo de la red aprovisionada o de los anfitriones que albergan los recursos virtuales.

A continuación se describirán algunas de las herramientas mas conocidas actualmente son descritas a continuación:

Fabric es una herramienta de automatización que usa SSH [8] para hacer despliegues de aplicaciones y administración de tareas. Fabric es una librería gratuita hecha en python y su forma de interactuar es por medio de linea de comandos. Por otra parte permite cargar y descargar archivos que pueden ser ejecutados por su conjunto de funciones [4].

Chef es una de las herramientas más conocidas de automatización de infraestructura de nube, esta escrita en Ruby y Erlang. Utiliza un lenguaje de dominio especifico, expresado también en Ruby para la escritura y configuracion de los recursos que deben ser creados, a esto se le denomina recetas". Estas recetas contienen los recursos que deben ser creados. Chef se puede integrar con plataformas basadas en la nube, como Rackspace, Internap, Amazon EC2, Cloud Platform Google, OpenStack, SoftLayer y Microsoft Azure. Adicionalmente puede aprovisionar sobre contenedores si se instala la librería indicada.

Chef contiene soluciones para sistemas de pequeña y gran escala [10].Es uno de los cuatro principales sistemas de gestión de configuración en Linux, junto con Cfengine, Bcfg2 y Puppet. La version gratuita puede ser utilizada, pero no cuenta con todo el conjunto de caracteristicas, administracion y soporte brindados por la herramienta. Estos se pueden obtener pagando una licencia de uso.

Puppetes una herramienta diseñada para administrar la configuración de sistemas similares a Unix y a Microsoft Windows de forma declarativa. El usuario describe los recursos del sistema y sus estados utilizando el lenguaje declarativo que proporciona Puppet. Esta información es almacenada en archivos denominados manifiestos Puppet. Puppet descubre la información del sistema a través de una utilidad llamada Facter, y compila los manifiestos en un catalogo especifico del

[8]SSH (Secure SHell, en español: intérprete de órdenes seguro) es el nombre de un protocolo y del programa que lo implementa, y sirve para acceder a máquinas remotas a través de una red

sistema que contiene los recursos y las dependencias de dichos recursos, estos catálogos son ejecutados en los sistemas de destino [11]. Puppet es de uso gratuito para redes muy pequeñas de hasta solo 10 nodos.

Juju es una herramienta de configuración y administración de servicios en nubes publicas. Permite crear ambientes completos con unos pocos comandos, cuenta con cientos de servicios pre-configurados y disponibles en la tienda de juju. Se puede usar a través de una interfaz gráfica o de linea de comandos. Juju permite re-crear un ambiente de producción en portátiles usando contenedores enfocado a pruebas. El uso de juju es gratuito pero se debe pagar por el uso de la nube publica [5].

CFEngine es un sistema basado en el lenguaje escrito por Mark Burgess, diseñado específicamente para probar y configurar software. CFEngine es como un lenguaje de muy alto nivel. Su objetivo es crear un único archivo o conjunto de archivos que describen la configuración de cada máquina de la red. CFEngine se ejecuta en cada host, y analiza cada archivo (o archivos), que especifica una política para la configuración del sistema. La configuración de la máquina es verificada contra el modelo y, si es necesario, cualquier desviación de la configuración es corregida. [12]

CFEngine cuenta con una versión gratuita y la versión empresarial que cuenta con interfaz gráfica, soporte y reportes.

Ansible es una herramienta de código libre desarrollada en python y comercialmente ofrecida por AnsibleWorks los cuales la definen como un motor de orquestación muy simple que automatiza las tareas de despliegue. Ansible no usa agentes, solo necesita tener instalado Python en las máquinas hosts y las tareas las realiza por medio de ssh. Ansible puede trabajar mediante un solo archivo de configuración que contendría todo o por medio de varios archivos organizados en una estructura de directorios [8].

Bcfg2 esta escrito en Python y permite gestionar la configuración de un gran numero de ordenadores mediante un modelo de configuración central. Bcfg2 funciona con un modelo simple de configuración del sistema, modelando elementos intuitivos como paquetes, servicios y archivos de configuración (así como las dependencias entre ellos). Este modelo de configuración del sistema se utiliza para la verificación y validación de las máquinas, permitiendo una auditoria robusta de los sistemas desplegados. La especificación de la configuración de Bcfg2 está escrita utilizando un modelo XML declarativo. Toda la especificacion puede ser validada utilizando los validadores de esquema XML ampliamente disponibles. Bcfg2 no tiene soporte para contenedores. Es gratuito y cuenta con una lista limitada de plataformas en las cuales trabaja bien.[13]

Cobbler es una plataforma que busca el rápido despliegue de servidores y en general computadores en una infraestructura de red por medio de linea de comandos, se basa en el modelo de scripts y cuenta con una completa base de simples comandos, que permite hacer despliegues de manera rápida y con poca intervención humana. Cobbler es capaz de instalar máquinas físicas y máquinas virtuales. Cobbler, es una pequeña y ligera aplicacion, que es extremadamente facil de usar para pequeños o muy grandes despliegues. Es de uso gratuito y no cuenta con soporte para contenedores. [14]

SmartFrog es un framework para servicios de configuración, descripción, despliegue y administración del ciclo de vida de máquinas virtuales. Consiste de un lenguaje declarativo, un motor que corre en los nodos remotos y ejecuta plantillas escritas en el lenguaje de SmartFrog y un modelo de componentes. El lenguaje soporta encapsulación (que es similar a las clases de python), herencia y composición que permite personalizar y combinar configuraciones. SmartFrog, permite enlaces estáticos y dinámicos entre componentes, que ayudan a soportar diferentes formas de conexión en tiempo de despliegue.

El modelo de componentes, administra el ciclo de vida a través de cinco estados: instalado, iniciado, terminado y fallido. Esto permite al motor del SmartFrog detectar fallas y reiniciar automáticamente re-despliegues de los componentes [7].

SmartFrog es desarrollado y mantenido por un equipo de investigación en los laboratorios de Hewlett-Packard en Bristol, Inglaterra, así como por el laboratorio Europeo de Hewlett-Packard y por contribuciones de otros usuarios de SmartFrog y desarrolladores externos a HP. Se utiliza en la investigación de HP, específicamente en la automatización de la infraestructura y automatización de servicios, ademas de ser solo utilizado en determinados productos de HP.

Amazon EC2 es un API propietario de Amazon que maneja un enfoque manual, el cual permite desplegar imágenes de máquinas virtuales conocidas como AMI (Amazon Machine Images) [9], que son las imágenes que se utilizan en Amazon para arrancar instancias virtuales. El concepto de las AMIs [9] es similar a las máquinas virtuales de otros sistemas. Básicamente estan compuestas de una serie de ficheros de datos

[9]La AMI de Amazon es una imagen mantenida y compatible que ofrece Amazon Web Services para su uso en Amazon Elastic Compute Cloud (Amazon EC2).

que conforman la imagen y luego un xml que especifica ciertos valores necesarios para que sea una imagen valida para Amazon.

Docker compose permite describir un conjunto de contenedores que se relacionan entre ellos. Docker composer define una aplicación multicontenedor en un archivo con las mismas propiedades que se indicarían en un archivo individual de docker. Docker composer usa archivos en formato yaml para describir las características de los servicios en cada contenedor [6]. Es completamente gratuito.

Vagrant es una herramienta de linea de comando que permite la creación y configuración de entornos de desarrollo virtualizados. Originalmente se desarrolló para VirtualBox y sistemas de configuración tales como Chef, Salt y Puppet. Sin embargo desde la versión 1.1 Vagrant es capaz de trabajar con múltiples proveedores, como VMware, Amazon EC2, LXC y DigitalOcean [15].

Vagrant ofrece múltiples opciones para realizar aprovisionamientos, desde scritps en shell hasta complejos sistemas de configuración.

SaltStack es un sistema de manejo de configuración cuyo objetivo es garantizar que un servicio este corriendo o que una aplicación haya sido instalada o desplegada. Salt está construido en Python y al igual que Chef, CFEngine y Puppet utiliza un esquema de cliente (salt minions) - servidor (salt master), cuyo método de conexión con los minions se realiza a través de un broker messages [10] llamado ZeroMQ (0MQ), que no solo garantiza una conexión segura sino que la hace confiable y rápida. Salt tiene una versión gratuita llamada Salt Open sin embargo para obtener todos los beneficios se debe pagar la versión empresarial. Salt cuenta con soporte para contenedores. [16]

IV. Conclusion

The conclusion goes here.

Acknowledgment

The authors would like to thank...

Referencias

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.
[2] ACM Symposium on Cloud Computing 2015, http://acmsocc.github.io/2015, 2015.
[3] Andrew S. Tanembaum, Modern Operating Systems 4th Edition, 2014.
[4] Fabric, http://www.fabfile.org/, 2016.
[5] Ubuntu Juju, http://www.ubuntu.com/cloud/juju, 2016.
[6] Docker Composer, https://docs.docker.com/compose, 2016.
[7] SmartFrog, http://smartfrog.org/display/sf/SmartFrog+Home, 2009.
[8] Ansible, https://www.ansible.com, 2016.
[9] Amazon EC2, https://aws.amazon.com/ec2, 2016.
[10] Chef, https://docs.chef.io, 2015.
[11] Puppet, http://projects.puppetlabs.com/projects/puppet, 2015.
[12] Cfengine, https://www.gnu.org/software/cfengine, 2001.
[13] Bdfg2, http://bcfg2.org, 2015.
[14] Cobbler, https://cobbler.github.io, 2016.
[15] Vagrant, https://www.vagrantup.com, 2016.
[16] SaltStack, http://saltstack.com/community, 2016.
[17] Architectural Styles and the Design of Network-based Software Architectures, https://www.ics.uci.edu/ fielding/pubs/dissertation/top.htm, 2000.
[18] Inversion of Control Containers and the Dependency Injection pattern, http://martinfowler.com/articles/injection.html, 2004.
[19] HMAC: Keyed-Hashing for Message Authentication, https://www.ietf.org/rfc/rfc2104.txt, 1997.
[20] Linux Containers, https://linuxcontainers.org, 2016..
[21] LXC Templates Documentation, https://help.ubuntu.com/lts/serverguide/lxc.html, 2016.
[22] Docker, https://docker.com, 2016.
[23] REST - An Alternative to RPC for Web Services Architecture, http://web.uvic.ca/ērikj/seng422/resources/rest_paper.pdf, 2009.

[10] es un programa intermediario que traduce los mensajes de un sistema desde un lenguaje a otro, a través de un medio de telecomunicaciones.