Facultad 2

Trabajo de Diploma para optar por el título de
Ingeniero en Ciencias Informáticas

Diseño de arquitectura para herramientas de gestión de clientes basada en políticas

Autores: Jorge Luis Alfonso Chavez

Tutores: MsC. Mónica Peña Casanova
Ing. Yenlys Guerra Dávila

La Habana, 5 de junio de 2019

“Tengo un sueño, un solo sueño, seguir soñando”.

Martin Luther King

Declaración de autoría

Declaro ser autor de la presente tesis que tiene por título: “Diseño de arquitectura para herramientas de gestión de clientes basada en políticas” y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo. Para que así conste firmo la presente a los ___ días del mes de _____ del año 2019.

Jorge Luis Alfonso Chavez

_____

Firma del Autor

MsC. Mónica Peña Casanova

_____

Firma del Tutor

Ing. Yenlys Guerra Dávila

_____

Firma del Tutor

Datos de contacto

Tutor 1:

MsC. Mónica Peña Casanova

Graduada de Ingeniera en Telecomunicaciones, máster en ciencias y profesora auxiliar, imparte Teleinformática en la facultad 2 y actualmente se desempeña como decana en la misma.

Universidad de las Ciencias Informáticas

Email: monica@uci.cu

Tutor 2:

Ing. Yenlys Guerra Dávila

Graduada de Ingeniera en Ciencias Informáticas, profesora asistente, imparte Sistemas Operativos en la Facultad 2, Instructora CISCO y actualmente se desempeña como subdirectora del Centro de Telemática.

Universidad de las Ciencias Informáticas

Email: yguerra@uci.cu

Agradecimientos

En primer lugar, a mi familia por apoyarme todos los días durante estos 5 años de carrera.

Dedicatoria

A

Jorge y Cristina,

Heyda y Bebo, Elia y Chavez,

y a mi hombro de apoyo, Douglas.

Resumen

Las Tecnologías de la Información y las Comunicaciones evolucionan de manera que el desarrollo social en el planeta no está a la altura del actual desarrollo científico en el ámbito de la informática y las comunicaciones. Con este rápido avance aparecen los entornos heterogéneos, dispersos y con altos niveles de complejidad en la gestión de clientes de red dentro de las organizaciones. Estos entornos enmarcan los principales motivos por lo cual es necesario efectuar una correcta gestión de clientes de red a la hora de brindar los servicios, ya que de ello depende brindar un servicio con calidad. La gestión integrada de clientes de red, se presenta como una opción viable dado que permitirá el intercambio de información de gestión en entornos heterogéneos, a través de un modelo común de información vinculando las necesidades del negocio con la infraestructura de la organización.

En el presente trabajo se diseña una arquitectura de software para la gestión de clientes de red basado en políticas en entornos tecnológicos inestables que permite aumentar la disponibilidad, modificabilidad y usabilidad en las soluciones de gestión de clientes. Para el diseño de la arquitectura propuesta se utiliza el Método de Diseño Basado en la Arquitectura (ABD), se valida con el Método de Análisis de Acuerdos de Arquitectura (ATAM) utilizando escenarios para obtener la calidad de la arquitectura, también como forma de validación se realiza la implementación de funcionalidades para una herramienta de gestión de clientes de red.

Palabras clave: Arquitectura, cliente, gestión, política, red.

Abstract

The Information and Communication Technologies are evolving in such a way that social development on the planet is not up to the current scientific development in the field of informatic technology and communications. With this rapid advance appear heterogeneous environments, dispersed and with high levels of complexity in the management of network clients within organizations. These environments frame the main reasons why it is necessary to make a correct management of network clients when providing services, since it depends on providing a quality service. The integrated management of network clients is presented as a viable option given that it will allow the exchange of management information in heterogeneous environments, through a common information model linking the needs of the business with the infrastructure of the organization.

In the present work a software architecture is designed for the management of network clients based on policies in unstable technological environments that allows to increase the availability, modifiability and usability in the client management solutions. For the design of the proposed architecture the Architecture-Based Design Method (ABD) is used, it is validated with the Architecture Trade-off Analysis Method (ATAM) using scenarios to obtain the quality of the architecture, also as a validation form the implementation of functionalities is performed for a network client management tool.

Keywords: Architecture, client, management, network, policy.

Índice de contenido

Índice de tablas

Índice de figuras

Introducción

Las organizaciones son cada vez más dependientes de la tecnología para llevar a cabo sus objetivos y ejecutar sus procesos. Las informaciones críticas son almacenadas, procesadas y transmitidas en formato digital. El éxito de la introducción de las Tecnologías de la Información (TI) en las organizaciones está determinado por la gestión que se realiza de las mismas, a través de la habilitación de un conjunto de sus capacidades utilizando un grupo de facilitadores que forman parte de las prácticas de gestión. Por ello, los administradores de TI se enfrentan al desafío de: alinear (hacer converger, armonizar, integrar, enlazar, sincronizar) los objetivos y procesos de TI a los de las organizaciones; aplicar la mejora continua en los servicios que prestan, como habilitadores y conductores de los cambios en las organizaciones.

Para lograr este alineamiento en entornos de alta complejidad caracterizados por: dispositivos heterogéneos y dispersos, con cortos períodos de obsolescencia; múltiples servicios sometidos a variados requerimientos regulatorios; y usuarios que hacen uso de ellos desde contextos diversos, la automatización de la gestión de infraestructuras TI se ha convertido en un imperativo.

En las organizaciones uno de los dominios a gestionar de manera automatizada es el de gestión de clientes que se encarga de la configuración de sistemas clientes, específicamente del despliegue de sistemas operativos, la gestión de inventarios de hardware y software, la distribución de software, la gestión de parches, el monitoreo del uso de las aplicaciones y el control remoto de los dispositivos.

En la medida que las organizaciones tienen mayor cantidad de equipos interconectados es mayor la “superficie de ataque” y por tanto es necesario dar seguimiento a los parches y vulnerabilidades de una amplia gama de sistemas operativos y aplicaciones, convirtiendo a la informatización de este proceso en imponderable. La superficie del ataque es un término empleado en el contexto de la seguridad informática para denotar las posibilidades de ataque de un atacante informático.

El proceso de gestión de clientes de red requiere del establecimiento de gran cantidad de controles en entornos heterogéneos y de inmediatez en la respuesta ante los diferentes cambios en el entorno. Por otra parte, debe operar de manera coordinada con otros dominios de gestión, por ejemplo, el de equipamiento activo de redes, el de servicios y servidores, lo que hace que el mismo sea complejo y crítico.

Dada la problemática descrita anteriormente se plantea el siguiente **problema a resolver**: ¿Cómo alinear la gestión de clientes de red en entornos tecnológicos inestables?

Para dar solución a la problemática presentada se propone como **objeto de estudio**, el proceso de gestión de clientes de red.

Se trazó como **objetivo general** de la investigación: Diseñar una arquitectura de software para la gestión de clientes de red basado en políticas en entornos tecnológicos inestables.

El **campo de acción** se enmarca en: Arquitectura para la automatización de la gestión de clientes de red basado en políticas.

Las **preguntas científicas** que guían y orientan el proceso investigativo son:

- ¿Cómo se encuentra el estado actual de las arquitecturas de gestión de clientes de red?
- ¿Cuáles son las metodologías de diseño de arquitecturas de software?
- ¿Qué tecnologías existen en la actualidad para la gestión de clientes de red?
- ¿Cómo se diseña una arquitectura de software para herramientas de gestión de clientes de red?
- ¿Cómo validar la propuesta de solución a través de metodologías de validación de arquitecturas de software?

Para dar respuesta a las actividades trazadas durante la realización de este informe, se plantea el cumplimiento de las siguientes **tareas de la investigación**:

- Análisis de arquitecturas basadas en gestión de clientes de red y las principales características que incluyen estas, para la elaboración de un marco teórico de la investigación.
- Estudio de metodologías de diseño de arquitecturas de software para la selección de la arquitectura basada en sistemas de gestión de clientes de red.
- Selección de las herramientas, metodología y tecnologías a utilizar en el desarrollo de la arquitectura.
- Extracción de las entradas necesarias de los métodos escogidos de definición y validación de la arquitectura para el análisis de las mismas.
- Análisis del modelo ATAM (Método de Análisis de Acuerdos de Arquitectura) para validar la arquitectura propuesta.

La investigación está sustentada en los siguientes métodos científicos:

1. Métodos teóricos:
    - Histórico-Lógico para realizar un estudio cronológico a partir de la consulta de la bibliografía en gestión de redes e inventario de equipos.
    - Analítico-Sintético para analizar y comprender la documentación relacionada con la implantación de políticas en equipos de redes.
    - Modelación de los diferentes diagramas correspondientes al diseño de la arquitectura.
2. Métodos empíricos:
    - Encuesta para consultar a los especialistas en gestión de redes y servicios telemáticos, con el objetivo de obtener información que sustente la presente investigación (*ver Anexo 1*).

Estructura del documento

Para una mejor comprensión de la investigación, el contenido del presente trabajo de diploma se encuentra estructurado en 3 capítulos, conclusiones, recomendaciones, referencias bibliográficas, bibliografía y anexos. Los contenidos a tratar en cada capítulo son:

Capítulo 1: Fundamentación teórica. Se abordan diferentes temas relacionados con la gestión de redes y servicios, protocolos o tecnologías para gestión de clientes de red, sobre las políticas y estándares. Estado del arte sobre los diferentes estilos arquitectónicos existentes para gestión de clientes. Descripción de las herramientas, metodología y tecnologías a utilizar en el desarrollo de la arquitectura.

Capítulo 2: Propuesta de solución. Se brinda una explicación de la arquitectura propuesta, detallando cada uno de los artefactos de entrada, análisis y de salida utilizados en el diseño de la arquitectura según lo que propone el método de diseño seleccionado.

Capítulo 3: Se valida la arquitectura propuesta, mediante el método de validación seleccionado y las fases del mismo, así mismo se emplea el desarrollo de un software, otro método para realizar la ratificación de la misma en un entorno de trabajo real.

1. Capítulo I. Fundamentación Teórica

En el capítulo se realiza el estudio de los conceptos y definiciones relacionado con el tema de investigación, así como una descripción de la gestión de clientes y la gestión de clientes basada en políticas y de los distintos protocolos o tecnologías relacionados con el tema. También se realiza una descripción de las herramientas de trabajo utilizadas, así como las tecnologías y metodologías utilizadas para elaborar y validar la solución.

1.1. Definiciones de interés

1.1.1. Cliente de red

Se define en la presente investigación como cliente de red, a cualquier dispositivo que consuma servicios de los proveedores de una infraestructura de red determinada.

1.1.2. Política

Un conjunto de reglas que se utilizan para gestionar y mantener el control de los cambios y-o mantener el estado de uno o varios objetos gestionados, para controlar y coordinar, de manera dinámica los elementos de red, tomando decisiones de forma automática a través de reglas, peticiones de usuarios o de servicios, se emplea la PBNM (gestión de redes basada en políticas). (Baró Menéndez, Anias Calderón y Peña Casanova 2018)

1.1.3. Sistema Operativo (SO)

Un SO es el programa que, luego de ser cargado inicialmente por un programa de arranque (boot program), gestiona todos los otros programas en el equipo. Los otros programas son conocidos como aplicaciones. Las aplicaciones hacen uso del sistema operativo haciendo peticiones para servicios a través de una API (Interfaz de programa de aplicaciones) definida. Los usuarios pueden interactuar directamente con el sistema operativo usando una GUI (Interfaz gráfica de usuario) o la CLI (Interfaz de línea de comandos). (Rouse 2016)

1.1.4. Tarjeta de red (NIC)

Una NIC, es una placa de circuitos o tarjeta que es instalada en un ordenador, para que pueda estar conectado a una red informática (Rouse 2006). Aunque con el desarrollo actual, se pueden encontrar en otros dispositivos que son conectados a la red, dígase los clasificados como SMART[1] y otros que hacen uso de estas tecnologías para realizar sus funciones.

[1] Viene del término “inteligente” traducido del inglés. Un dispositivo inteligente es un dispositivo electrónico, por lo general conectado a otros dispositivos o redes a través de diferentes protocolos como Bluetooth, NFC, Wi-Fi, 3G, X10, etc, que puede funcionar hasta cierto punto de forma interactiva y autónoma.

1.1.5. Herramientas de gestión de clientes (CMT)

Las CMT[2], tratan las configuraciones de los sistemas clientes. Específicamente incluyen funcionalidades como: despliegue de sistemas operativos, inventario, distribución de software, gestión de parches, monitoreo y control remoto de uso de software (Gartner 2018).

1.1.6. Sistema de gestión de red (NMS)

Un NMS, es un sistema para el monitoreo, mantenimiento y optimización de una red de trabajo. Incluye tanto hardware como software, pero se conoce mayormente como un software usado para el mantenimiento de la red. (Christensson 2016)

1.1.7. Arquitectura de referencia

Una arquitectura de referencia está formada por un documento o un conjunto de documentos que ofrecen estructuras e integraciones recomendadas de productos y servicios de TI para formar una solución. La arquitectura de referencia incorpora las mejores prácticas aceptadas del sector, que normalmente sugieren el método de entrega o las tecnologías concretas óptimas. (HPE 2019)

1.1.8. Servicio TI

Los servicios TI se refieren a la aplicación de experiencia comercial y técnica para permitir que las organizaciones creen, gestionen y optimicen o accedan a la información y los procesos empresariales (Gartner 2017). El “cliente" y el "proveedor" pueden referirse a una pareja tal como aplicación-aplicación, abonado-aplicación, abonado-operador y operador-operador.

1.1.9. Servicio Web

Son servicios[3] que están disponibles desde el servidor web de una empresa para usuarios web u otros programas conectados a la web. (Rouse 2007)

1.2. Gestión de clientes

Es la planificación, organización, supervisión y control de los elementos que forman una red y del rango de capacidades suministradas para garantizar un nivel de servicio de acuerdo a un costo. Pretende optimizar la pertinencia, disponibilidad y rendimiento de los clientes y capacidades suministradas e incrementar su efectividad alcanzándose una mayor productividad en la institución y un aumento de la satisfacción de los usuarios. Los sistemas de gestión que existen actualmente, utilizan una estructura básica, conocida por paradigma gestor-agente, cuyo esquema queda reflejado en la Figura 1. (Millán Tejedor 1999)

[2] Previamente conocidas como Herramientas de control de ciclo de vida de ordenadores, PCCLM por sus siglas en inglés PC configuration life cycle management.
[3] Generalmente incluyen una combinación de programación, datos y recursos humanos.

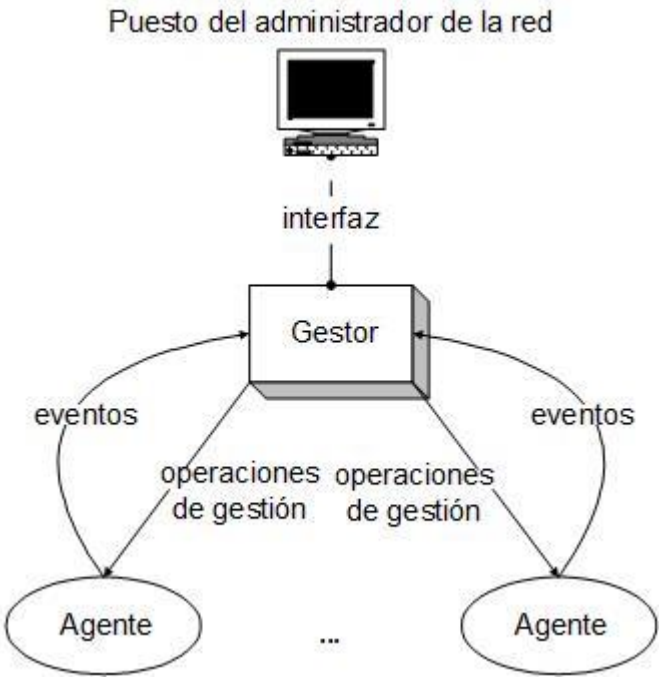

Figura 1 Paradigma Gestor-Agente

Los sistemas de apoyo a la gestión se componen por lo general de:

- Interfaz con el operador o el responsable de la red. Esta interfaz a la información de gestión, a través de la cual el operador puede invocar la realización de operaciones de control y vigilancia de los recursos que están bajo su responsabilidad, es una pieza fundamental en la consecución de un sistema de gestión que tenga éxito. Se puede componer de alarmas y alertas en tiempo real, análisis gráficos y reportes de actividad.
- Elementos hardware y software repartidos entre los diferentes componentes de la red.

Los elementos del sistema de gestión de clientes, bajo el paradigma **gestor-agente**, se clasifican en dos grandes grupos:

- Los **gestores** son los elementos del sistema de gestión que interaccionan con los operadores humanos y desencadenan acciones necesarias para llevar a cabo las tareas por ellos invocadas.
- Los **agentes**, por otra parte, son los componentes del sistema de gestión invocados por el gestor o gestores de la red.

El principio de funcionamiento reside en el intercambio de información de gestión entre nodos gestores y nodos gestionados. Habitualmente, los agentes mantienen en cada nodo gestionado información acerca del estado y las características de funcionamiento de un determinado recurso de la red. El gestor pide al agente, a través de un protocolo de gestión, que realice determinadas

operaciones con estos datos de gestión, gracias a las cuales podrá conocer el estado del recurso y podrá influir en su comportamiento. (Millán Tejedor 1999)

Cuando se produce alguna situación anómala en un recurso gestionado, los agentes, sin necesidad de ser invocados por el gestor, emiten los denominados eventos o notificaciones que son enviados a un gestor para que el sistema de gestión pueda actuar en consecuencia.

Existen distintos protocolos de gestión de redes (protocolos de gestión integrada), dentro de los cuales destaca SNMP[4] (Protocolo de gestión de red simple) perteneciente al conjunto de protocolos TCP/IP. Este es el protocolo a utilizar en redes empresariales, pues todos los equipos lo soportan y, de hecho, SNMP puede ser considerado el estándar de facto. Otro protocolo estándar, es el protocolo común de gestión de información (CMIP), de la familia de protocolos del modelo OSI[5] de la ISO[6].(Millán Tejedor 1999)

Según (Millán Tejedor 1999), el modelo de gestión ISO clasifica las tareas de los sistemas de gestión en cinco áreas funcionales. La tarea del encargado de gestionar una red empresarial será evaluar la plataforma de gestión a utilizar en cuanto a la medida en que dicha plataforma resuelva la problemática de gestión en cada una de estas áreas:

- **Gestión de configuración:** Obtener datos de los clientes y utilizarlos para incorporar, mantener y retirar los distintos componentes y recursos a integrar.
- **Gestión de rendimiento:** Mantenimiento del nivel de servicio que el cliente ofrece a sus usuarios, asegurándose de que está operando de manera eficiente en todo momento.
- **Gestión de contabilidad:** Medir parámetros de utilización del cliente que permitan a su explotador preparar las correspondientes facturas a sus usuarios.
- **Gestión de fallos:** Localización y recuperación de los problemas.
- **Gestión de seguridad:** Ofrecer mecanismos que faciliten el mantenimiento de políticas de seguridad.

Las herramientas o plataformas de gestión de clientes se encargan de la recepción de informes y datos mediante el sondeo automático o iniciado por el usuario, a diferentes dispositivos de la red. En el caso de reconocer algún problema en dichos parámetros, las entidades de gestión las notificarán al operador, almacenarán los eventos e intentarán reparar el sistema automáticamente.

La gestión remota de ordenadores se ha vuelto importante en el soporte de hardware y software. Las posibilidades de asistencia directa, aumentan debido al incremento de conexiones de clientes a

[4] Es un marco para la gestión de sistemas en un entorno de red. Se puede usar para la configuración y el control basados en políticas utilizando un Módulo MIB específico diseñado para ejecutar políticas en elementos administrados mediante scripts. Los elementos en un dispositivo de red se evalúan utilizando un filtro de políticas, para determinar dónde se aplicará la política.

[5] Acrónimo de Open Systems Interconnection

[6] Acrónimo de International Organization for Standarization, empresa dedicada a realizar estándares para el uso de equipos, protocolos, software y otras implementaciones de objetos y documentos.

través de internet, la configuración de la intranet interna de las compañías y los métodos de telecomunicaciones convencionales.

La gestión remota puede ayudar al equipo de soporte con el uso de programas. Si un programa *gestor* se encuentra instalado en el ordenador del administrador, este tiene acceso a una gran cantidad de configuraciones en cualquier equipo con una aplicación *agente* instalada.

De mano con el inventario, la distribución de software, el despliegue de sistemas operativos y la gestión de parches, la gestión remota es una de las herramientas más importantes para una efectiva gestión de clientes; porque no solo se tiene que conocer el error y la forma de localizarlo, sino que también puede ser fácilmente solucionado desde el área de trabajo del administrador de la red.

1.3. Estilos arquitectónicos para soluciones basadas en redes

Se realiza el estudio de diferentes estilos arquitectónicos que se emplean en el diseño de soluciones basadas en redes. Los mismos son evaluados siguiendo los parámetros propuestos por (Fielding 2000), que son los que debe cumplir una arquitectura dentro de sus características:

- **Rendimiento:** El rendimiento de una aplicación basada en la red está limitado primero por los requisitos de la aplicación, luego por el estilo de interacción elegido, seguido por la arquitectura realizada y por la implementación de cada componente.
- **Escalabilidad:** La escalabilidad se refiere a la capacidad de la arquitectura para admitir una gran cantidad de componentes o interacciones entre componentes, dentro de una configuración activa. La escalabilidad se puede mejorar al simplificar los componentes, al distribuir servicios a través de muchos componentes y al controlar las interacciones y configuraciones como resultado del monitoreo. Los estilos influyen en estos factores al determinar la ubicación del estado de la aplicación, la extensión de la distribución y el acoplamiento entre los componentes.
- **Simplicidad:** El principal medio por el cual los estilos arquitectónicos inducen simplicidad es mediante la aplicación del principio de separación de preocupaciones a la asignación de funcionalidad dentro de los componentes. Si se puede asignar la funcionalidad de tal manera que los componentes individuales sean sustancialmente menos complejos, entonces serán más fáciles de entender e implementar. Del mismo modo, dicha separación facilita la tarea de razonar sobre la arquitectura general.
- **Modificabilidad:** La modificabilidad se refiere a la facilidad con la que se puede realizar un cambio en la arquitectura de una aplicación. Una preocupación particular de los sistemas basados en red es la modificabilidad dinámica, donde la modificación se realiza en una aplicación implementada sin detener y reiniciar todo el sistema.
- **Visibilidad:** Los estilos también pueden influir en la visibilidad de las interacciones dentro de una aplicación basada en la red al restringir las interfaces a través de la generalidad o proporcionar acceso al monitoreo. La visibilidad en este caso se refiere a la capacidad de un componente para monitorear o mediar la interacción entre otros dos componentes. La

visibilidad puede permitir un mejor rendimiento a través del almacenamiento en caché compartido de las interacciones, la escalabilidad a través de los servicios en capas, la confiabilidad a través del monitoreo reflexivo y la seguridad al permitir que los mediadores inspeccionen las interacciones.

- **Portabilidad:** El software es portátil si se puede ejecutar en diferentes entornos. Los estilos que inducen la portabilidad incluyen aquellos que mueven el código junto con los datos a procesar y aquellos que restringen los elementos de datos a un conjunto de formatos estandarizados.

- **Confiabilidad:** La confiabilidad, dentro de la perspectiva de las arquitecturas de aplicaciones, se puede ver como el grado en que una arquitectura es susceptible de fallar en el nivel del sistema en presencia de fallas parciales dentro de los componentes, conectores o datos. Los estilos pueden mejorar la confiabilidad al evitar puntos únicos de falla, habilitar la redundancia, permitir el monitoreo o reducir el alcance de la falla a una acción recuperable.

Existen diferentes estilos arquitectónicos para el desarrollo de aplicaciones siguiendo la filosofía de gestión de clientes de red, estos se encuentran divididos en grupos. Están los estilos jerárquicos, de flujo de información, basados en réplicas, también se encuentran los estilos de código móvil y los muy conocidos Peer-to-Peer (P2P). De los grupos anteriores se toma una parte de los estilos existentes en los jerárquicos y los P2P, ya que están más enfocados al desarrollo de aplicaciones para la gestión de clientes de red.

- **Cliente-servidor (CS):** el más frecuente de los estilos arquitectónicos para aplicaciones basadas en red. Un componente de servidor, que ofrece un conjunto de servicios, escucha las solicitudes de dichos servicios. Un componente cliente, que desea que se realice un servicio, envía una solicitud al servidor a través de un conector. El servidor rechaza o realiza la solicitud y envía una respuesta al cliente.

- **Cliente-servidor por capas (LCS)**: agrega componentes de proxy y puerta de enlace al estilo cliente-servidor. Un proxy actúa como un servidor compartido para uno o más componentes del cliente, tomando solicitudes y enviándolas, con posible traducción, a los componentes del servidor. Un componente de puerta de enlace parece ser un servidor normal para los clientes o proxies que solicitan sus servicios, pero de hecho está reenviando esas solicitudes, con posible traducción, a sus servidores de "capa interna". Estos componentes mediadores adicionales se pueden agregar en varias capas para agregar funciones al balanceo de carga y verificación de seguridad en el sistema.

- **Cliente-servidor sin estado (CSS):** deriva de cliente-servidor con la restricción adicional de que no se permite ningún estado de sesión en el componente del servidor. Cada solicitud del cliente al servidor debe contener toda la información necesaria para comprender la solicitud y no puede aprovechar ningún contexto almacenado en el servidor. El estado de la sesión se mantiene enteramente en el cliente.

- **Sesión remota (RS):** variante de cliente-servidor que intenta minimizar la complejidad o maximizar la reutilización, de los componentes del cliente en lugar del componente del servidor. Cada cliente inicia una sesión en el servidor y luego invoca una serie de servicios en el mismo, finalmente saliendo de la sesión. El estado de la aplicación se mantiene enteramente en el servidor. Este estilo se usa normalmente cuando se desea acceder a un servicio remoto utilizando un cliente genérico o mediante una interfaz que imita a un cliente genérico.

- **Objetos distribuidos (DO):** organiza un sistema como un conjunto de componentes que interactúan como iguales. Un objeto es una entidad que encapsula cierta información o datos privados del estado, un conjunto de operaciones o procedimientos asociados que manipulan los datos y posiblemente un hilo de control, de modo que colectivamente se puedan considerar una sola unidad. En general, el estado de un objeto está completamente oculto y protegido de todos los demás objetos. La única forma en que se puede examinar o modificar es mediante una solicitud o invocación en una de las operaciones de acceso público del objeto. Esto crea una interfaz bien definida para cada objeto, lo que permite que la especificación de las operaciones de un objeto se haga pública y al mismo tiempo mantenga la implementación de sus operaciones y la representación de la información de su estado en privado, mejorando así la capacidad de evolución.

Se procede a relacionar cada estilo seleccionado con los parámetros definidos anteriormente, para extraer una posible vía de diseño de la solución. En la simbología a utilizar en la tabla se tiene el signo negativo (-) y positivo (+), los cuales marcan una influencia negativa y positiva respectivamente, la celda vacía es para los casos que no tengan influencia en el parámetro. La tabla a continuación muestra una adaptación a los estilos arquitectónicos seleccionados según las métricas que propone (Fielding 2000) con estadísticas actualizadas.

Tabla 1 Impacto de los estilos arquitectónicos sobre las soluciones finales

Estilo	Rend.	Escalab.	Simp.	Modific.	Visib.	Portab.	Confiab.	Total
CS		+	+	+				3
LCS	-	++	+	++		+		5
CSS		++	+	+	+		+	6
RS	+	-	+	+	-			1
DO	-			++++	-		-	1

Como puede ser observado, el estilo que posee mayor impacto en los atributos medidos es el “cliente-servidor sin estado”, por su gran contribución a la escalabilidad de las soluciones que poseen dicho estilo, así como la simplicidad, modificabilidad, visibilidad y confiabilidad que aporta.

1.4. Tecnologías para la gestión de clientes de red

Para la gestión remota de los clientes, se usan diversas tecnologías o combinaciones de las mismas para garantizar disponibilidad de los dispositivos a administrar. Son mayormente utilizados por las

empresas en el control de los activos, ya sea para monitorizar las actividades de los empleados o para instalar características en los mismos y tener el software de los ordenadores en su última versión. Dentro de estas tecnologías, existe un grupo que no requieren de la interacción con los usuarios en las estaciones de trabajo y poseen características que son utilizadas por muchas organizaciones en la realización de tareas de soporte y despliegue de software. Algunas de estas son:

1.4.1. Wake on LAN (WOL)

Tecnología de hardware y software que permite el accionamiento de un ordenador apagado mediante el envío de un paquete especial a un ordenador que esté equipado y habilitado para responderlo. La tecnología permite a los administradores realizar mantenimiento en el sistema, incluso si el usuario ha apagado el equipo. De esta forma, la característica permite al administrador conectar sistemas remotamente (siempre que sea la misma red local) para que puedan actualizarse o, si se configuró previamente, para iniciar una máquina con un servidor de acceso remoto para que un cliente de acceso remoto puede realizar tareas en este sistema (Cavalcante et al. 2018). WoL cuenta con múltiples variantes para las diferentes NIC que existen en los dispositivos, las cuales van desde *Wireless WoL* (WWoL), para las tarjetas inalámbricas en los equipos y para los clientes que utilicen tecnologías móviles (EDGE, GPRS, 3G, 4G, 5G, entre otras).

1.4.2. Entorno de ejecución prearranque (PXE)

PXE se define sobre la base de los protocolos de Internet estándar de la industria y los servicios que son ampliamente utilizados en la industria, es decir, TCP/IP, DHCP (protocolo de configuración dinámica de host) y TFTP (protocolo de transferencia de archivos trivial). Estos estandarizan la forma de las interacciones entre clientes y servidores. Para garantizar que el significado de la interacción cliente-servidor también esté estandarizado, se utilizan ciertos campos de opción de proveedor en el protocolo DHCP, que están permitidos por el estándar DHCP. Las operaciones de los servidores DHCP y/o BOOTP estándar (Protocolo de Bootstrap), que brindan direcciones IP y/o NBP[7], no se verán interrumpidas por el uso del protocolo extendido. Los clientes y servidores que conozcan estas extensiones (utilizadas para identificar las peticiones procedentes de clientes que implementan el protocolo PXE) reconocerán y utilizarán esta información y los que no reconocen las extensiones las ignorarán. (Intel Corporation y SystemSoft 1999)

1.4.3. Redes privadas virtuales (VPN)

Una VPN extiende una red privada a través de una red pública y permite a los usuarios enviar y obtener información a través de redes agrupadas o públicas como si sus maniobras informáticas estuvieran directamente asociadas al sistema enclaustrado. Por lo tanto, las aplicaciones que se ejecutan en la VPN pueden beneficiarse de la funcionalidad, la seguridad y la administración de la red privada. (Karuna Jyothi y Reddy 2018)

[7] Acrónimo para Network Bootstrap Program. La imagen de arranque remota descargada por el cliente PXE vía TFTP o Multicast TFTP.

1.5. Gestión de clientes basada en políticas

Según (Kosiur 2001) la gestión de basada en políticas, está estructurada por una arquitectura de gestión de red basada en políticas. Los elementos que la componen en su diseño original son los siguientes:

- **Consola de políticas:** Sirve como interfaz entre el administrador de la red y el resto del sistema de gestión de políticas. Esta trabaja en conjunto con la herramienta de gestión de políticas para traducir las reglas de políticas que el administrador de la red crea o edita, en una forma que sea compatible con el esquema del repositorio de políticas. En la mayoría de los usos de estas arquitecturas el protocolo para acceder a los datos del repositorio es LDAP.

- **Herramienta de gestión de políticas:** herramienta para la creación y edición de las políticas y luego almacenarlas en el repositorio que las almacena. Permite la vista general de las políticas, la traducción de las políticas de un lenguaje de modelado a archivos de configuraciones que comprendan los PEP. También valida las políticas y resuelve los conflictos globales que pueden surgir a la hora de implantarlas en los PEP.

- **Repositorio de políticas:** Directorio que almacena las políticas creadas o editadas por el administrador de la red con la herramienta de gestión de políticas. Posee características para la búsqueda y retorno de las mismas.

- **Punto de decisión de política:** este convierte las políticas definidas por el administrador y las independientes del dispositivo en políticas dependientes del dispositivo que los puntos de ejecución de políticas admitan. El PDP también juega el rol de resolver los conflictos entre políticas. También poseen mecanismos para comunicación con los PEP para determinar cuándo una política no fue cargada satisfactoriamente, forzando una reversión a una política implantada anteriormente.

- **Proxy de políticas:** se considera como un módulo traductor que interviene entre los PDP y los dispositivos de la red. Es el responsable de convertir las políticas generadas por los PDP en archivos de configuración que los dispositivos no consientes de políticas puedan comprender y ejecutar. Esto significa que mayormente el proxy traducirá las políticas en comandos CLI (líneas de comandos) o SNMP.

- **Punto de ejecución de política:** utiliza los comandos que recibe de los PDP (directamente o por un proxy de políticas) para procesarlos. La forma en que un PEP lo procesa depende totalmente del tipo de dispositivo que sea (switch, router, ordenadores, dispositivos móviles)

Según el grupo de trabajo del Internet (IETF), un modelo de gestión basado en políticas, incluye un contenedor o repositorio de políticas, un punto de decisión de políticas (PDP) o servidor de políticas y uno o varios puntos de ejecución de políticas (PEP).

En los PEP se aplican o ejecutan las políticas que gobiernan los dispositivos físicos. El agente PDP revisa las políticas almacenadas en el contenedor de políticas y efectúa un proceso de toma de

decisiones. El PDP envía las decisiones tomadas, que son independientes de las características de los dispositivos a los PEP asociados. Estos PEP se encargan de traducirlas en operaciones o comandos específicos que puedan ser interpretados por la tecnología concreta de los agentes que actúan en los recursos gestionados por ellos. La figura a continuación muestra una arquitectura general para un sistema PBNM siguiendo la filosofía cliente-servidor.

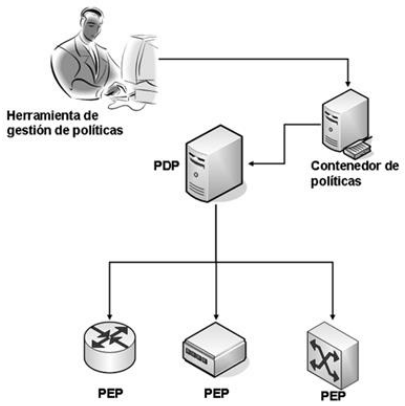

Figura 2 Arquitectura PBNM propuesta por el IETF.

La mayor parte de los sistemas se gestionan en una ventana de tiempo, de ahí que la gestión basada en políticas sea una arquitectura efectiva para lograr este objetivo. La definición de políticas en la PBNM, se ha trabajado de dos maneras, con la tupla condición-acción, la cual tiene problemas para la capacidad de predicción (por cuanto no se pueden ejecutar políticas hasta que las condiciones que se evalúan son visibles o percibirles en la red y los servicios) y la eficiencia (porque chequear constantemente la ocurrencia de condiciones implica un costo computacional alto) y la tríada evento-condición- acción, esta última permite al sistema determinar cuándo serán evaluadas las condiciones. Los eventos de políticas representan los estados del sistema que son relevantes en el contexto de los objetivos de negocio y su realización operacional. Las acciones de políticas son las respuestas deseadas por la organización en caso de que ocurra uno o más eventos de políticas. Las reglas de políticas son los mecanismos que enlazan los eventos de políticas con las acciones de políticas. (Baró Menéndez, Anias Calderón y Peña Casanova 2018)

1.6. Consideraciones de las herramientas de gestión de clientes (CMT)

Según los especialistas y analistas de Gartner, organización de investigación industrial, los CMT poseen 8 funciones críticas, las cuales forman una solución completa de gestores de clientes.

- Gestión de inventarios
    - Dispositivos escritorio con Sistema Operativo
    - Dispositivos escritorio sin Sistema Operativo
    - Dispositivos móviles
- Gestión de parches
- Escalabilidad

- Extensibilidad
- Gestión de sistemas operativos
- Gestión de herramientas de software
    - Distribución
    - Monitoreo y control de uso
- Administración de sistemas
- Soporte remoto

Las presentes funciones permiten identificar las características principales que deben presentar las soluciones de gestión de clientes, acotando así la investigación con precisión y evidenciar la complejidad y justificar el cumplimiento de la arquitectura a diseñar con las funcionalidades ofertadas por las soluciones de un CMT.

1.7. Métodos, herramientas y tecnologías

En el desarrollo de la investigación, se emplearon diferentes herramientas, métodos y tecnologías para la validación, diseño y desarrollo de la solución.

1.7.1. Métodos para el diseño de arquitecturas

En el sentido de aplicaciones y proyectos de software, existen diversos métodos para diseñar, documentar y evaluar las arquitecturas de software, entre los más destacados se pueden encontrar:

- **Lenguaje Unificado de Modelado (UML):** es un lenguaje de modelado visual que se emplea para especificar, visualizar, construir y documentar los artefactos de un sistema de software (Larman 2004). Se ha convertido en un estándar para los sistemas de software, gestionado por el Object Management Group, conocido como OMG (Traver Salcedo 2002). Su objetivo es lograr modelos que, además de escribir con cierto grado de formalismo tales sistemas, puedan ser entendidos por los clientes o usuarios de aquello que se modela. (Rumbaugh, Jacobson y Booch 2007)

- **Modelo de las 4+1 vistas:** es un modelo de visión diseñado por Philippe Kruchten, describe la arquitectura del software usando cinco vistas concurrentes. Cada vista se refiere a un conjunto de intereses de los diferentes stakeholders (notación para hacer mención de los clientes en el modelo 4+1 vistas) del sistema (Kruchten 1995). Kruchten agrupa estas 5 vistas por su naturaleza en 3 apartados; el conceptual donde sitúa a la vista lógica y la de procesos, el físico que lo componen las vistas de componentes y la distribuida y por último la funcional la que se refiere a la vista de casos de uso.

- **Método de Diseño Basado en la Arquitectura (ABD):** es un método de desarrollo en el que el diseño de arquitecturas se basa en los requisitos de calidad y que sigue una filosofía top-down (de arriba hacia abajo) a partir de una descomposición funcional del problema y una división del sistema en subsistemas más simples, eligiendo a cada paso el estilo arquitectónico más adecuado. En cada iteración del proceso de diseño, se realiza un proceso

de evaluación para determinar si se cumplen los requisitos establecidos. (Bachmann et al. 2000)

- **Estilos de Arquitectura basada en Atributos (ABAS):** son soluciones basadas en estilos arquitectónicos que han surgido de la experiencia de resolver problemas que se presentan de manera frecuente. ABAS utiliza los atributos de calidad para definir un marco de aplicación de un estilo arquitectónico concreto, proporcionando un razonamiento, cuantitativo o cualitativo, que fundamente la utilización de un estilo arquitectónico en un diseño determinado. (Klein et al. 1999)

- **Método de Análisis de Acuerdos de Arquitectura (ATAM):** es un método de evaluación de arquitectura de software desarrollado e impulsado por el SEI (Instituto de Ingeniería de Software), está inspirado en tres áreas distintas, los estilos arquitectónicos, el análisis de atributos de calidad y el Método de Análisis de Arquitecturas de Software (SAAM). Es una técnica que permite analizar arquitecturas de software con el objetivo de validar requerimientos de atributos de calidad, su interacción (conocidos como tradeoffs), detectar problemas de manera temprana e identificar riesgos y puntos sensibles. Se concentra en la identificación de los estilos arquitectónicos o enfoques arquitectónicos utilizados sobre la base de la técnica de escenarios. El método comprende nueve pasos, agrupados en las fases presentación, investigación y análisis, pruebas e informes. (Kazman, Klein y Clements 2000)

En la presente investigación se utilizarán dos de los métodos o estilos analizados anteriormente. Para el diseño de la propuesta de solución de la presente investigación, se utiliza ABD, ya que se basa principalmente en requisitos de calidad mediante la descomposición funcional del problema.

Para la evaluación y encuesta de satisfacción final del producto, ATAM es el método a utilizar, ya que está enfocado principalmente en esta tarea. Provee salidas que se refieren a parámetros de calidad, uso y rendimiento de arquitecturas mediante el uso de patrones, estilos o enfoques arquitectónicos en los escenarios descritos en su desarrollo.

1.7.2. Tecnologías

Para el desarrollo de un software que permita validar en un entorno real la solución de la investigación se utilizan las siguientes tecnologías.

- **C++:** Actualmente el lenguaje C++ es utilizado en muchos programas que necesitan rapidez y las funciones de bajo nivel que brinda este. Estas funciones permiten trabajar directamente con el Hardware, pudiendo así tener un control más profundo sobre las funcionalidades de estos (Medvidovic y Taylor 2000).

- **Qt:** Aunque Qt fue originalmente desarrollado para ayudar a los programadores de C++, se encuentran disponibles enlaces para diferentes lenguajes como: Java, JavaScript, PHP, Python, Ruby y la plataforma .NET. Dentro de sus características principales, se encuentran las señales y los slots, potentes para la simplicidad de la llamada a métodos desde clases distintas sin necesidad de invocarlos. También posee librerías muy eficaces para el trabajo

con los sistemas operativos, hardware, manejo de cadenas intuitivo y soporte para bases de datos relacionales basadas en SQL. También posee una estructura para el modelado de las GUI, el cual permite que puedan ser compiladas en cualquier sistema operativo que soporte el framework sin necesidad de modificarle parámetros o atributos a la misma. (Company 2016)

1.7.3. Herramientas

Se tiene en cuenta diversas herramientas para facilitar el desarrollo del software para la validación en entorno real de la solución, así como herramientas para el diseño de los artefactos del método.

- **Git:** Software de control de versiones diseñado por Linus Torvalds, pensando en la eficiencia y la confiabilidad del mantenimiento de versiones de aplicaciones cuando éstas tienen un gran número de archivos de código fuente. (Chacon y Straub 2014)

  La principal diferencia entre Git y cualquier otro VCS (incluyendo Subversion y semejantes) es la forma en la que manejan sus datos. Conceptualmente, la mayoría de los otros sistemas almacenan la información como una lista de cambios en los archivos. Git no maneja ni almacena sus datos de esta forma. Git maneja sus datos como un conjunto de copias instantáneas de un sistema de archivos miniatura.

  La mayoría de las operaciones en Git sólo necesitan archivos y recursos locales para funcionar. Por lo general no se necesita información de ningún otro computador. Todo en Git es verificado mediante una suma de comprobación antes de ser almacenado y es identificado a partir de ese momento mediante dicha suma. Cuando realizas acciones en Git, casi todas ellas solo añaden información a la base de datos de Git (Chacon y Straub 2014). Git tiene tres estados principales en los que se pueden encontrar los archivos: confirmado (committed), modificado (modified) y preparado (staged).

- **GitLab:** Servicio web de control de versiones y desarrollo de software colaborativo basado en Git. Como gestor de repositorios, el servicio ofrece alojamiento de wikis y un sistema de seguimiento de errores, todo ello publicado bajo una Licencia de código abierto. Fue escrito por los programadores ucranianos Dmitriy Zaporozhets y Valery Sizov en el lenguaje de programación Ruby. La compañía, GitLab Inc., cuenta con un equipo de 630 miembros(GitLab 2019b) y más de 2200 usuarios(GitLab 2019a).

- **Qt Creator:** es un IDE multi plataforma programado en C++, JavaScript y QML creado por Trolltech[8] el cual es parte de SDK para el desarrollo de aplicaciones con GUI con las bibliotecas Qt. (Company 2016)

  Qt Creator se centra en proporcionar características que ayudan a los nuevos usuarios de Qt a aprender y aumentar la productividad. Posee un editor de código con soporte para C++, QML y ECMAscript, herramientas para la rápida navegación del código, resaltado de sintaxis y autocompletado de código, control estático de código y estilo a medida que se escribe,

[8] Compañía de software fundada en el año 1994 en Oslo, Noruega.

soporte para la refactorización de código, ayuda sensitiva al contexto, plegado de código (code folding), paréntesis coincidentes y modos de selección. (Qt Foundation 2018)

- **Visual Paradigm:** Herramienta CASE[9] utilizada para diseñar e implementar software informático. Permite el modelado de sistema utilizando UML y otros lenguajes. También permite personalizar las arquitecturas para las empresas, la gestión de proyectos, el uso de metodologías ágiles, el diseño de una interfaz para mostrar prototipos de la experiencia del usuario final, la mejora de los negocios a través de los diagramas de procesos de negocios y las simulaciones de los mismos, la edición de diagramas en entornos web y su capacidad para compartir estos, generador de códigos y bases de datos y posee además soporte para el trabajo en equipos. (Visual Paradigm 2015)

1.8. Metodología

Se usa la metodología que propone el "Diseño Basado en Arquitecturas", ABD, no para proponer una arquitectura de un sistema concreto sino para llevar a cabo una que sirva de guía para el desarrollo de soluciones de herramientas de gestión de clientes basadas en políticas. Para la concepción o diseño de la arquitectura se seguirán los pasos o las fases propuestos por el método ABD.

Los métodos de diseño como el método ABD no pretenden reemplazar a los diseñadores expertos. En su lugar, están destinados a apoyar a estos diseñadores al proporcionar una estructura dentro de la cual el diseño puede proceder. El método ABD proporciona una estructura simple y poderosa. Es simple porque el método se puede describir como un procedimiento recursivo con unos pocos pasos dentro de la recursión. Es potente porque proporciona un método de diseño que puede cumplir todos los requisitos y validar que se han cumplido.

El ABD puede inscribirse dentro del Proceso de Desarrollo Basado en la Arquitectura (Architecture Based Development Process) definido por Bass y Kazman en (Bachmann et al. 2000). Dicho proceso se resume en la figura a continuación. En ella puede observarse que el proceso de diseño se encuentra entre la definición de requisitos y el análisis de la arquitectura. El proceso puede repetirse de forma iterativa, resolviendo en cada iteración una parte del diseño.

[9] Acrónimo de Computer-aided software engineering, que significa Ingeniería Informática de Software.

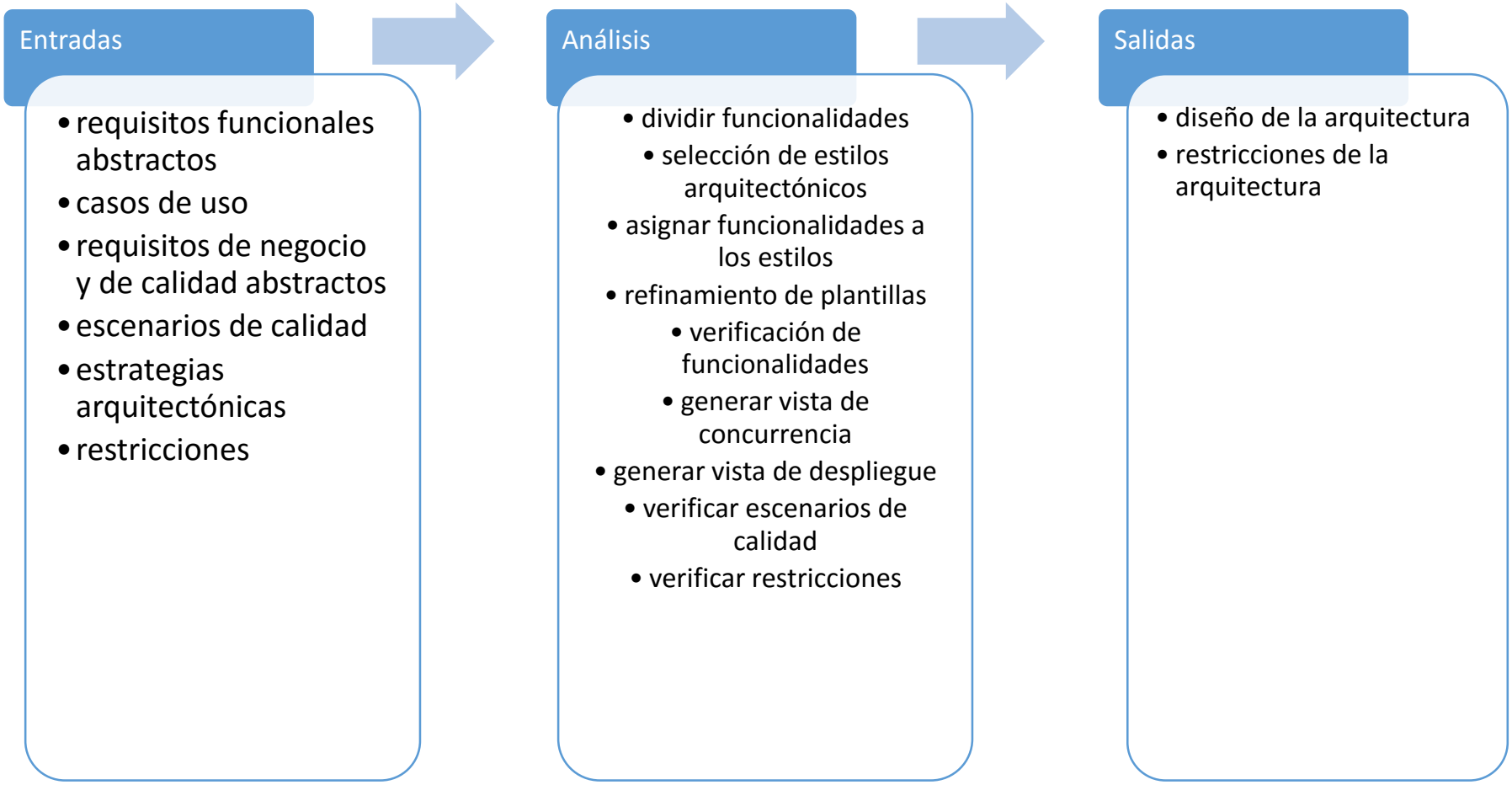

Figura 3 Ciclo de vida del Método ABD

Según (Bachmann et al. 2000), las entradas al método ABD son una lista de requisitos. Estos requisitos son funcionales (tanto abstractos como concretos), de calidad y de negocios (abstractos y concretos), como de restricciones. Un requisito es un elemento para el cual el diseñador tiene libertad para elegir una solución, una restricción es un elemento que especifica la decisión que debe tomar un diseñador. Los requisitos abstractos se utilizan para generar el diseño; Los requisitos concretos se utilizan para validar las decisiones tomadas como resultado de los requisitos abstractos.

1.9. Conclusiones del capítulo

Luego de realizado un marco teórico de la investigación se llegan a las siguientes conclusiones. La gestión de clientes en las redes es importante dentro del ámbito empresarial, ya que permite tener un control sobre los recursos que maneja cada uno de estos en su uso, para la selección de los estilos arquitectónicos en el diseño de soluciones basadas en redes, debe considerarse la modificabilidad, el rendimiento, la disponibilidad o confiabilidad y la simplicidad, entre otros atributos de calidad. Una gestión basada en políticas informáticas permite ahorro en tiempo y costo en las tareas que son hechas por los técnicos o administradores de sistemas en las empresas a la vez que facilita la automatización del control de los clientes en las redes, pero posee insuficiencias para la integración con otros dominios de gestión . Existen numerosos métodos para el diseño de arquitecturas, pero ABD se enfoca directamente en el desarrollo de estas, partiendo de los atributos de calidad de las mismas. Una correcta validación de las soluciones de este tipo, se puede realizar utilizando el método para las validaciones de arquitecturas ATAM, quedando como salida del mismo, una vista de la arquitectura y las restricciones que tiene la misma. Con los- artefactos definidos se puede realizar correctamente la implementación de las soluciones.

2. Capítulo II Propuesta de solución

En el presente capítulo se analizan las características de la arquitectura a diseñar, partiendo de su diseño lógico y la elaboración de los métodos y artefactos definidos por las metodologías y herramientas a utilizar.

2.1. Entradas del método ABD

Dentro de las estradas de ABD, se encuentran diferentes paquetes que son estructurados en una perspectiva general, esta se conoce como vista lógica dentro de la metodología en sí.

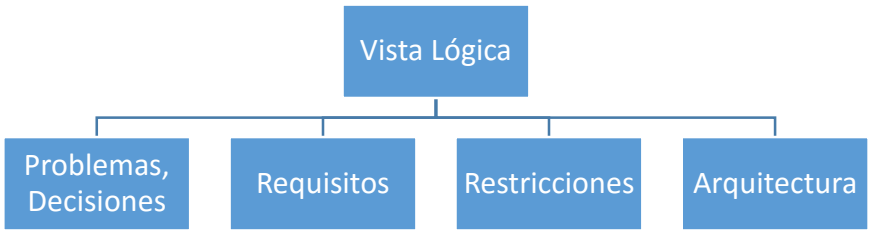

Figura 4 Estructura de paquetes de ABD

2.1.1. Casos de uso

En los siguientes diagramas de caso de uso de sistema, se muestran los requisitos funcionales que tienen que cumplir los CMT, en sus versiones agente, instaladas en los equipos que se necesitan sean controlados; y en su versión gestor, los controladores de los clientes, para monitorizar y gestionar los mismos.

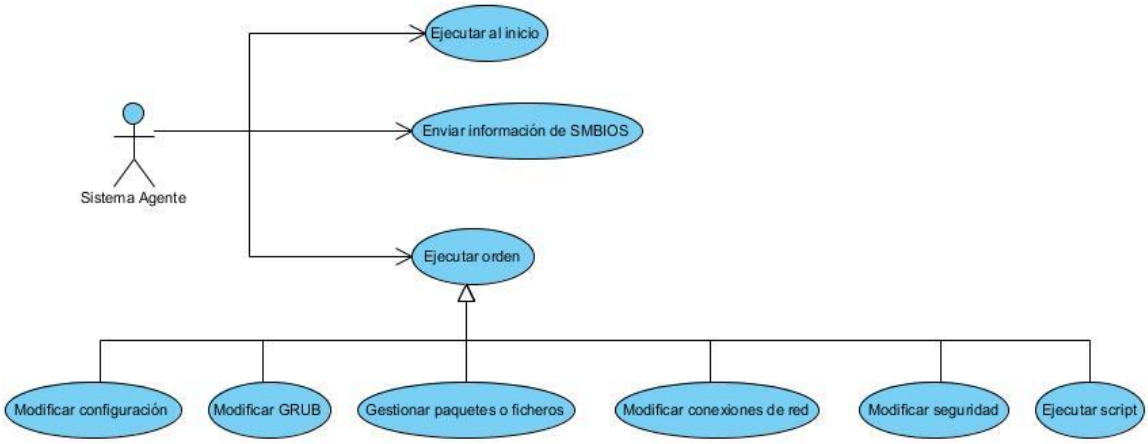

Figura 5 Diagrama de caso de uso del Sistema Agente

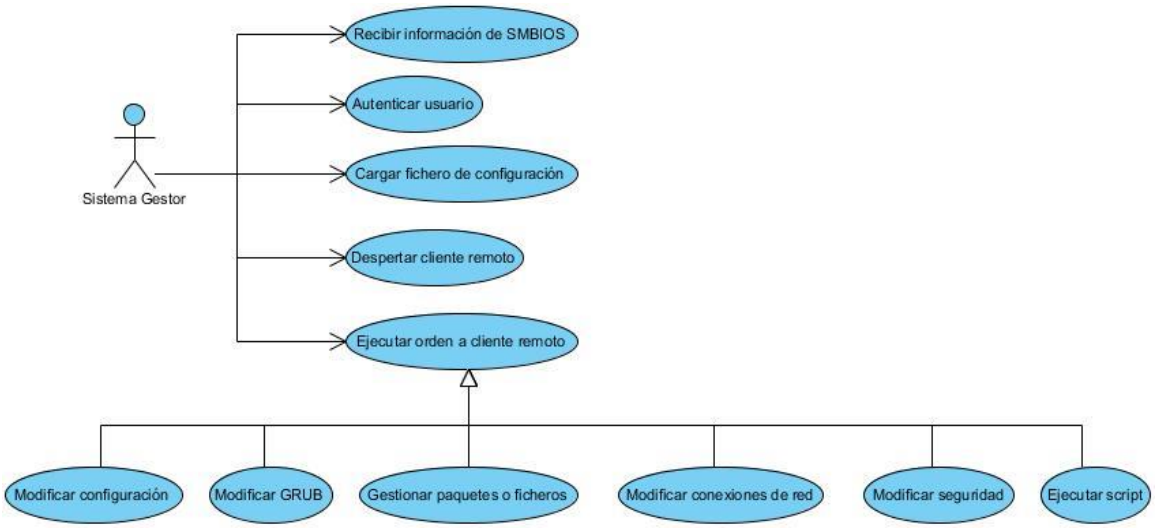

Figura 6 Diagrama de caso de uso del Sistema Gestor

2.1.2. Factores del negocio

2.1.2.1. Mercado potencial

Las soluciones de gestión de clientes constituyen una forma de mantener monitorizadas y controladas las tecnologías informáticas presentes en las organizaciones. Debido a la variedad de equipos de conexión presentes en las empresas actuales, es necesario tener un método efectivo para el control y extracción de datos, que a su vez sea personalizable para cada organización y/o empresa que haga uso de la solución.

2.1.2.2. Objetivos de la organización

La Universidad de las Ciencias Informáticas (UCI), como centro docente productor que dirige la formación a distancia, genera un conjunto importante de objetos de aprendizaje, así como productos y servicios informáticos. Su modelo de desarrollo de software, se encuentra certificado con la norma CMMI Nivel 2 lo que implica el almacenamiento de un conjunto de elementos que conforman los expedientes de proyectos. Por otra parte, el proceso de formación de pregrado o postgrado también se encuentra certificado, lo que implica un uso intensivo de su infraestructura TI y el almacenamiento de una cantidad importante de evidencias. Por todo lo anterior, una parte importante del patrimonio de la UCI, se encuentra en formato digital, de ahí que sea necesario para realizar diversas tareas con los equipos que almacenan esta información, que va desde instalación de parches de seguridad, actualizaciones del sistema, u otras modificaciones que se requieran necesarias hacer.

2.1.3. Requisitos funcionales abstractos

Los requisitos funcionales suelen estar más ligados a los sistemas que a la arquitectura, puesto que expresan cuales deben ser las funcionalidades que se le exige a un sistema concreto. No obstante, como los requisitos funcionales suelen ser numerosos, pueden ser organizados en diferentes niveles de abstracción, siendo concretados en las funcionalidades de un sistema. En consecuencia, se proponen en este apartado una lista de requisitos funcionales abstractos que pueden brindar la mayoría de las soluciones CMT:

- Gestión de seguridad
- Gestión de configuraciones
    - Red
    - Sistema
    - Usuario
    - Software
    - Recursos
- Gestión de inventario
- Gestión de paquetes
- Gestión de usuarios y control de acceso
- Gestión de disponibilidad de recursos

2.1.4. Requisitos de calidad abstractos

2.1.4.1. Aspectos de modificabilidad

Se refieren a la habilidad para hacer cambios al sistema de una forma rápida y poco costosa durante su desarrollo y luego su despliegue, estableciendo un balance entre el costo de construcción dentro del costo y el costo de cambio. Este atributo es crucial para la presente investigación ya que el objetivo de una línea de producción de software como el CMT, es reducir el tiempo, esfuerzo, costo y complejidad de crear y mantener los productos de la línea. Se puede agrupar en los aspectos

relacionados con la extensibilidad, portabilidad, reestructurabilidad, reusabilidad, interoperabilidad y mantenibilidad que presenta un componente o solución en la arquitectura a desarrollar, centrándose en los siguientes asuntos de interés:

- Probabilidad del cambio.
- Magnitud y dimensión.
- Costo del cambio.
- Complejidad del cambio.
- Conservación del conocimiento de la solución.
- Confiabilidad del cambio.

Teniendo en cuenta lo anteriormente planteado, la arquitectura debe satisfacer los siguientes requisitos:

- Asegurar que un componente con responsabilidades comunes (alta cohesión de responsabilidades) no tengan que depender demasiado de componentes con distintas responsabilidades (bajo acoplamiento entre componentes).
- Evitar modificar demasiados módulos al aparecer algún cambio.
- Proveer integración entre los componentes mediante interfaces abstractas, con el objetivo de poder sustituir un componente por otro, con la menor cantidad de cambios posibles.
- Proveer componentes reutilizables de forma tal que puedan ser empleados por otras soluciones.

2.1.4.2. Aspectos de rendimiento

El rendimiento expresa cual es el comportamiento del sistema o componente, al ejecutar una funcionalidad específica dentro de ciertas restricciones de velocidad, precisión, entre otras; así como en qué grado utiliza eficientemente los recursos. Los aspectos de rendimiento están muy relacionados con los demás atributos de calidad, por eso es vital tener una balanza entre todos los parámetros, de forma tal que no se contradigan unos y otros, logrando establecer una medida promedio entre todos. Su análisis se establece en cada uno de los componentes y en la arquitectura en general teniendo en cuenta los aspectos:

- Velocidades de la red y saturación.
- Cantidad de paquetes enviados contra paquetes recibidos.
- Tiempos de procesamiento.
- Respuesta ante fallos de software.
- Respuesta ante múltiples solicitudes.

Dada las particularidades del dominio, el rendimiento sólo puede caracterizarse con un muy alto grado de abstracción, ya que en dependencia de cada una de las funciones que caracterizan a los gestores de clientes será la calidad de la gestión y de la información recibida de los equipamientos en la red organizacional. Teniendo en cuenta lo anteriormente planteado la arquitectura debe satisfacer los siguientes requisitos:

- Establecer límites de tamaño y cantidad de los paquetes enviados en las subredes.
- Proporcionar mecanismos para establecer múltiples conexiones en los equipos monitorizados.
- Optimizar los procesos realizados para ser ejecutados en el menor tiempo posible.

2.1.4.3. Aspectos de seguridad

La seguridad es la medida sobre la habilidad del sistema para resistir usos no autorizados, mientras sigue proveyendo sus servicios a los usuarios legítimos. De forma genérica se puede caracterizar sobre la base de los siguientes aspectos:

- No repudio.
- Confidencialidad.
- Integridad.
- Disponibilidad.
- Garantía.
- Auditoría.

En las soluciones de CMT, la seguridad es un aspecto de criticidad alta, ya que puede verse comprometida información sensible, obtener acceso total a los equipos de la red y se puede interceptar, bloquear o modificar cualquier dato enviado. Es por ello que la arquitectura debe proveer los requisitos siguientes.

- Cifrar el contenido enviado a través de la red.
- Comprobación de la autoría de la información enviada.
- Almacenamiento y copias de seguridad de los datos.
- Poseer un registro de los cambios realizados.
- Poseer un mecanismo de control de acceso por roles.

2.1.4.4. Aspectos de disponibilidad

La disponibilidad recoge los aspectos que garantizan la probabilidad de que la solución esté operativa en un período de tiempo determinado. Por el momento es suficiente con caracterizar la disponibilidad mediante la identificación de los servicios o funciones más críticas de los sistemas y enumerar las estrategias y mecanismos que debe proporcionar la arquitectura para que dichos servicios estén disponibles y muestren un comportamiento fiable. En general, dichas estrategias y mecanismos se basan en aplicar diversas formas de redundancia homogénea o heterogénea. Los aspectos a destacar de este atributo son:

- Tiempo de recuperación.
- Disponibilidad de los datos.
- Soporte en varios sistemas operativos.
- Trabajo de manera remota.

Por lo antes expuesto la arquitectura debe proveer los siguientes requisitos:

- Proveer mecanismos de control remoto.
- Proveer mecanismos de almacenamiento y respaldo de los datos.
- Proporcionar mecanismos de recuperación ante fallos.
- Proveer la disponibilidad de los servicios brindados todos los días a toda hora.

2.1.4.5. Aspectos de usabilidad

Es la medida con la que un producto se puede usar por usuarios determinados para conseguir objetivos específicos con efectividad, eficiencia y satisfacción en un contexto de uso concreto. Es un atributo difícil de caracterizar incluso en sistemas concretos, pues depende de muchos factores, algunos de ellos ligados a la arquitectura y otros no. Este atributo puede ser subdividido en cinco propiedades específicas, las cuales son: características de aprendizaje del sistema para asistir a los usuarios en caso de que estos no conozcan alguna de sus características operativas; la posibilidad de usar un sistema eficientemente, esto es qué puede hacer el sistema para hacer más eficiente su operación; la capacidad de minimizar el impacto de errores, lo que sistema provee para que un error del usuario tenga el mínimo impacto; la adaptación del sistema a la necesidades del usuario; y por último lo que el sistema pueda hacer para aumentar la confidencialidad y satisfacción de los usuarios.

Teniendo en cuenta los aspectos expresados se tienen en cuenta los siguientes requisitos:

- Retroalimentar al usuario sobre las funciones que realiza el sistema.
- Proveer la posibilidad de cancelar y deshacer las acciones.
- Proveer funcionalidades y acciones identificables y de fácil acceso.

2.1.4.6. Aspectos de verificabilidad

Es el grado de facilidad que tiene un sistema para ser probado en su completitud, mediante pruebas de unidad, integración, aceptación, regresión, entre otras. Hace referencia a la facilidad con la cual el software puede ser construido para posteriormente encontrar defectos o problemas directamente probando sus componentes. Por lo antes expuesto la arquitectura debe proporcionar los siguientes requisitos:

- Proveer mecanismos de separación de interfaz e implementación y el uso de componentes específicos para la realización de pruebas.
- Proveer componentes o software especializado que permitan monitorizar el comportamiento de los componentes durante su ejecución.

2.1.5. Restricciones

Para garantizar un diseño correcto de la arquitectura propuesta, se tienen en cuenta ciertas restricciones, para el despliegue de la misma. Estas son redactadas por expertos en tecnologías de la información y comunicaciones.

Tabla 2 listado de restricciones de entrada al método ABD

Restricción	Atributo	Descripción
R1	Disponibilidad	El sistema debe funcionar las 24 horas de los 7 días de la semana durante todo el año.
R2	Disponibilidad	La integración entre los diferentes dominios que lo conforman.
R3	Seguridad	Se debe brindar solución de conflictos en las políticas de TI que se ejecuten en cada dominio.
R4	Usabilidad	Los sistemas operativos de los ordenadores son: Windows y GNU/Linux
R5	Seguridad	No se permiten las comunicaciones entre equipos de subredes distintas, con excepción de las subredes de administración de redes y servicios, que deben ser accesibles desde toda la infraestructura

2.1.6. Estrategias arquitectónicas

Los estilos arquitectónicos son un conjunto de descripciones de patrones e interacciones, algunos ejemplos de estos componentes, para el caso del dominio CMT son: sistemas operativos, aplicaciones de escritorio, configuraciones, ordenadores, actualizaciones, entre otros. En función de los estilos arquitectónicos identificados, se realiza la descomposición en elementos de diseño en el dominio CMT para el diseño de políticas de TI.

Teniendo en cuenta las consideraciones para las herramientas CMT, los factores del negocio, los requerimientos funcionales, las directrices, los atributos de calidad y las restricciones, se han seleccionado un conjunto de disposiciones arquitectónicas las cuales garantizan el éxito de la arquitectura.

Tabla 3 Listado de estrategias arquitectónicas escogidas

No	Estrategia arquitectónica
A1	Modelo de diseño de tablas de decisión
A2	Encapsulamiento basado en el modelo común de información (CIM/XML)
A3	Control de acceso basado en roles

A4	Backup server
A5	Mecanismo centralizado de gestión de trazas
A6	Patrón de implementación de políticas basado en GRASP
A7	Control de acceso basado en roles
A8	Arquitectura de repositorios
A9	Implementación de firmas digitales
A10	Uso de protocoles de transmisión seguros
A11	Basada en la puesta en marcha por ficheros de configuración
A12	Arquitectura con un mecanismo centralizado de gestión de errores e historial
A13	Arquitectura basada en eventos
A14	Arquitectura de tiempo real
A15	Arquitectura basada en estados

2.1.7. Conductores de la arquitectura

Los conductores de la arquitectura son los que permiten la implementación de las políticas de negocio en las diferentes soluciones técnicas a través de la estratificación de políticas, además facilitará la creación de políticas en un Punto de Decisión de Políticas Principal (PDPP) que permita la interacción de varios dominios de gestión de políticas, para tener un control holístico de la infraestructura desplegada. En este caso se propone el empleo del modelo DEN-ng para coordinar la comunicación entre los PDP de los diferentes dominios de gestión y el PDPP, aprovechando la definición de políticas con la tríada evento-condición- acción.

2.2. Análisis del método ABD

Según (Bachmann et al. 2000), el método ABD procede mediante la descomposición recursiva del sistema o los sistemas que se diseñarán. La primera descomposición es la del sistema; las descomposiciones posteriores son refinamientos de la descomposición previa. En cada paso de descomposición, los requisitos funcionales se cumplen asignando responsabilidades a las divisiones del elemento que se está descomponiendo. La calidad y los requisitos comerciales para ese

elemento se cumplen al elegir un estilo de arquitectura, basado en la calidad de conducción y los requisitos comerciales, que describe cómo se relacionan las divisiones entre sí.

La descomposición se examina desde la perspectiva de tres vistas: lógica, concurrencia y despliegue. Cada vista conducirá a responsabilidades adicionales para el elemento que está siendo descompuesto y estas responsabilidades deben ser capturadas en la descomposición.

Los requisitos concretos (funcionales, de calidad y de negocios) se utilizan para verificar las decisiones tomadas durante la descomposición. Las restricciones se examinan para verificar que ninguna de ellas haya sido violada por decisiones tomadas durante la descomposición.

2.2.1. Definición de la vista lógica

Se realizan los pasos para la definición de la vista lógica: descomposición de la funcionalidad, selección del estilo arquitectónico básico, asignación de funcionalidad a los componentes definidos en el estilo, refinamiento de las plantillas y verificación con casos de uso y escenarios de cambio. La división de la funcionalidad permite descomponer la misma en diferentes grupos según los criterios que dicten las directrices de la arquitectura, las cuales determinan además el estilo arquitectónico mediante el que deben organizarse los elementos resultantes de la descomposición. Dicho estilo arquitectónico define una serie de componentes a los cuales debe asignárseles cada uno de los grupos de funcionalidad resultantes de la descomposición. (Pastor Franco 2002)

2.2.1.1. Descomposición de las funcionalidades

Cada elemento de diseño tiene asignadas unas responsabilidades para satisfacer sus requisitos de diseño. Estas responsabilidades, que están constituidas por la funcionalidad que realiza el elemento y los atributos de calidad que le son propios, pueden dividirse en distintos grupos que pueden ser asignados a los elementos de diseño resultantes de su descomposición. Los criterios de división vienen determinados por los atributos de calidad que debe satisfacer el elemento de diseño.

Tabla 4 Listado de funcionalidades

No	Funcionalidad
F1	Leer información del SMBIOS de los clientes de red.
F2	Modificar las configuraciones de los clientes.
F3	Despertar cliente remoto por Wake-on-LAN.
F4	Establecer políticas en los equipos de una subred o individuales.
F5	Instalar salvas a determinados ficheros y/o paquetes en los equipos.
F6	Remover salvas a determinados ficheros y/o paquetes en los equipos.

F7	Copiar determinados ficheros y/o paquetes en los equipos.
F8	Realizar salvas a determinados ficheros y/o paquetes en los equipos.
F9	Modificar parámetros de seguridad.
F10	Recibir respuesta de los dispositivos.
F11	Ejecutar orden en una subred.
F12	Copiar archivo a cliente remoto.
F13	Modificar las configuraciones de red y otros medios de conexión.
F14	Ejecutar script remoto.
F15	Incluir una imagen de sistema en el GRUB.
F16	Autenticación utilizando un servidor LDAP.

Luego de asignar las responsabilidades a los subsistemas expertos en cada una de las funcionalidades, la vista lógica queda estructurada de la siguiente forma luego de una primera iteración del método ABD:

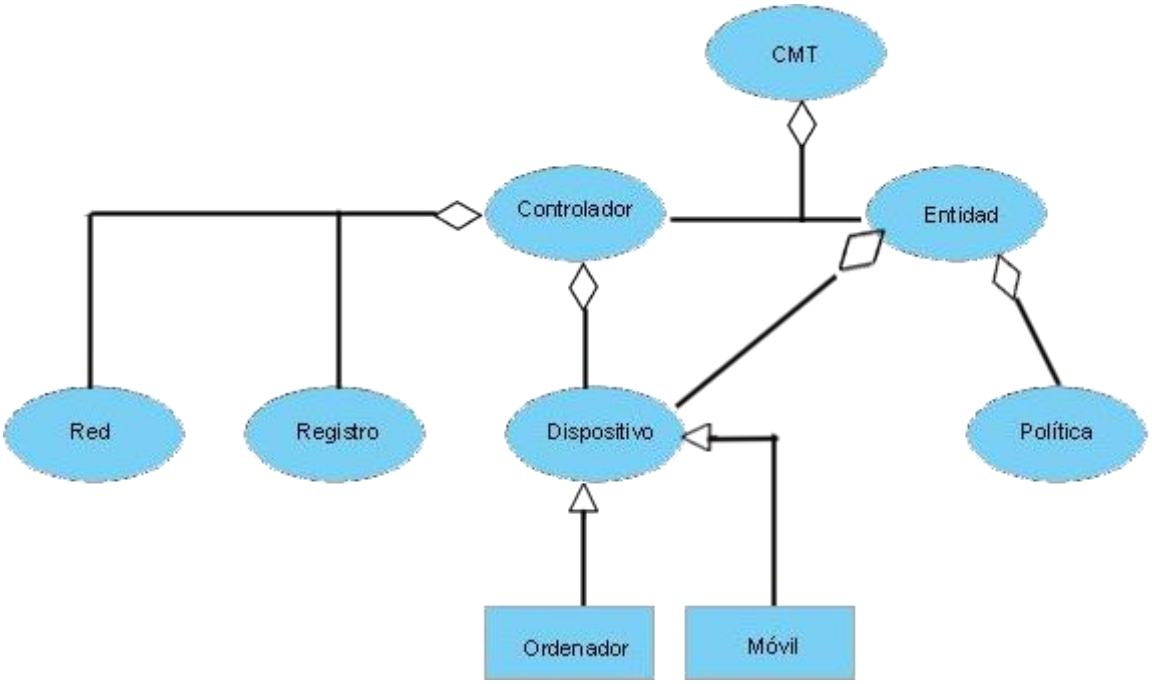

Figura 7 Vista lógica de la primera iteración de ABD

Es necesario realizar una segunda iteración, ya que se detecta que la entidad política, no cuenta con una plantilla genérica para las funciones que necesitan realizar los CMT, dígase los requisitos funcionales abstractos provistos en las entradas del método, quedando la misma de la siguiente forma:

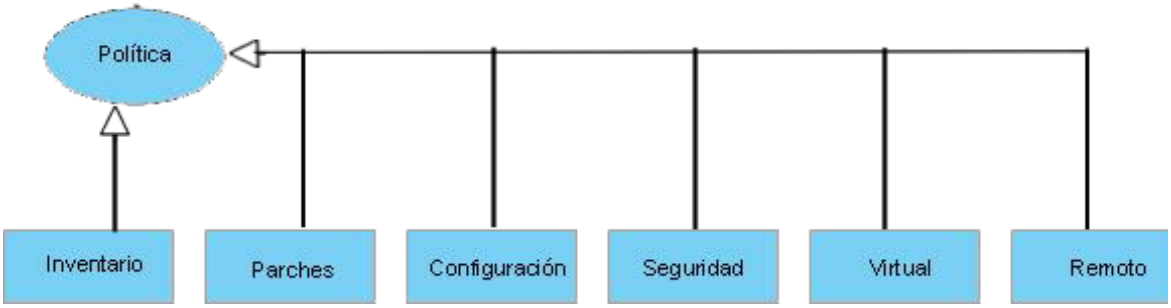

Figura 8 Vista lógica luego de la segunda iteración de ABD

Quedando estructurada totalmente la entidad política, se procede a unificar las vistas para definir la vista lógica de la arquitectura, quedando el CMT como principal elemento de la arquitectura ubicado en el primer subsistema, luego en un segundo están los controladores y entidades. Estas poseen elementos en un subsistema inferior para el tratamiento de la red y métodos de autenticación, registro de fallos e historial de comandos ejecutados y la gestión de los dispositivos, que puede ser considerados ordenadores o móviles. También se encuentran las políticas, clasificadas en 6 tipos, inventario, parches, configuración, seguridad, de entornos virtuales y de acceso remoto.

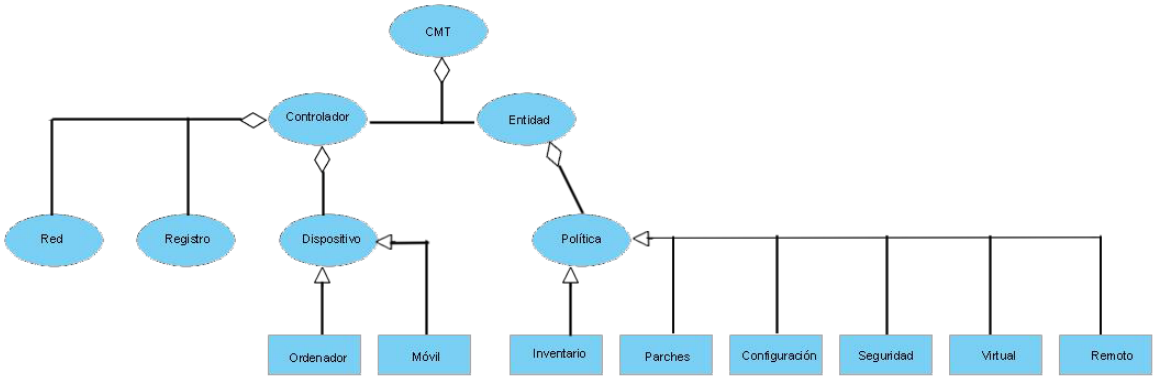

Figura 9 Vista lógica para una herramienta de gestión de clientes

2.3. Propuesta de solución para el despliegue del CMT

Para el despliegue del CMT se propone el empleo de la arquitectura se propone a la arquitectura PBNM definida por el IETF modificada. La modificación consiste en incorporar un PDPP (Punto de decisión de Políticas Principal), el cual tiene una jerarquía superior y puede detectar y resolver conflictos entre políticas, según lo explicado en el componente diseño de políticas, forzando a los PDP a regresar a los PEP a un estado anterior, en caso que se manifieste una condición prevista como, por ejemplo, la ocurrencia de un conflicto entre las políticas que se ejecutan en las capas jerárquicas inferiores. La arquitectura modificada empleará los modelos de información CIM (Modelo Común de Información) del DMTF (Grupo de Trabajo de Gestión Distribuida) entre los PDP y los PEP permitiendo la operación de la arquitectura tanto en soluciones de gestión integrada, como propietarias(Casanova y Calderón 2018); y del modelo de información DEN-ng entre el PDPP y los PDP, para facilitar la representación de políticas a través de la triada evento-condición-acción y conceder al sistema la posibilidad de determinar cuándo serán evaluadas las condiciones, otorgándole al mismo capacidad de respuesta temprana, ya que no es necesario que las condiciones sean visibles para que se ejecuten las acciones(Strassner 2004). Asimismo, hacer esto, garantiza mayor eficiencia en el funcionamiento de la arquitectura porque es posible definir eventos a partir de los cuales se evaluarán las condiciones, lo que implica no tener que invertir recursos computacionales y de ancho de banda adicionales para la evaluación periódica de las condiciones. También se incorpora a la arquitectura propuesta por el IETF, una base de datos de configuración, CMDB (por las siglas en inglés del Configuration Management Database, *ver anexo 7*) para la evaluación de condiciones y registro de las configuraciones que se modifiquen como resultado de la ejecución de políticas en los PEP.

Para ejecutar las acciones de gestión se utiliza el PDP, especializado en tomar las decisiones de cuáles políticas se deben ejecutar y del procesamiento de las mismas, alineando de esta forma las necesidades del negocio con el comportamiento coherente de la infraestructura TI. Para este fin, el PDP transforma las políticas o reglas en representaciones operacionales aptas para ser interpretadas por PEP que se encuentran en los elementos gestionados. El PDP debe contar con un motor de inferencia y se basa, para la toma de decisiones, en la información contenida en la CMDB y en el Repositorio de Políticas.

Luego de estructurado, modelado y diseñado las conexiones entre los elementos de la arquitectura propuesta para dar solución a la investigación, se elabora la vista de despliegue de la misma.

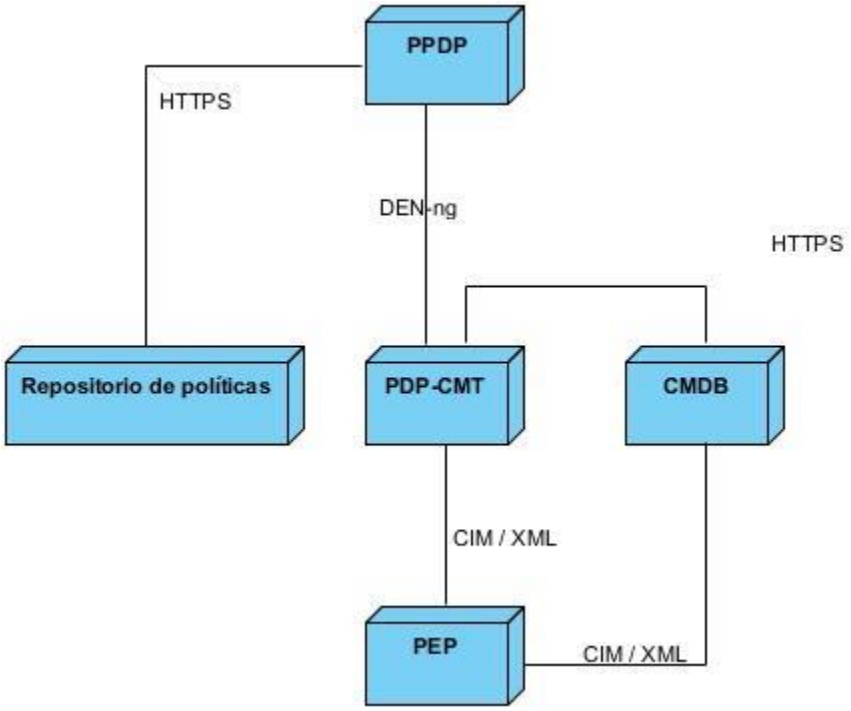

Figura 10 Vista de despliegue de la arquitectura

2.4. Conclusiones del capítulo

Luego de analizadas las entradas para el método ABD, se llega al modelo final que será sometido a las validaciones y pruebas pertinentes en las etapas de comprobación de ATAM. La extracción de los atributos de calidad y funcionales abstractos, así como los requisitos funcionales, haciendo cumplir las restricciones brindadas, deja una guía para las arquitecturas de los sistemas que utilicen un CMT dentro de sus herramientas de administración. Aplicando una descomposición funcional se llega a asignar a cada elemento de la arquitectura una responsabilidad a cumplir, haciendo uso de las iteraciones para satisfacer en cada una de ellas la implementación de los requisitos funcionales abstractos, cumpliendo las restricciones.

3. Capítulo III Validación

En el presente capítulo se realiza la validación, elemento indispensable para la comprobación de una investigación de carácter científico. Se ejecuta la misma mediante ATAM, método para la validación de arquitecturas de software, utilizando diferentes escenarios diseñados para verificar la capacidad de alineamiento de la arquitectura propuesta. Se establece como otra forma de validación, la implementación de un CMT evaluando la respuesta a algunos escenarios que son descritos en los arboles de decisión de la arquitectura realizados por ATAM.

3.1. Aplicación del método de validación ATAM

3.1.1. Atributos

Actualmente la UCI cuenta en el escenario CMT con la herramienta de gestión de inventarios GRHS, se aplicó un instrumento a un grupo 7 de expertos (*Ver Anexo 6*) para evaluar un conjunto de métricas de calidad en la herramienta actual, arrojando los resultados siguientes:

Tabla 5 Resultados de la aplicación de la técnica de grupo focal

Atributo	Estado
Modificabilidad	70%
Rendimiento	90%
Disponibilidad	80%
Seguridad	90%
Usabilidad	90%

Se procede a la construcción del árbol de utilidad de la arquitectura propuesta, aplicada el dominio de ordenadores. El árbol de utilidad contiene los escenarios que ponen a prueba a la arquitectura PBNM modificada propuesta en el capítulo anterior. Se construyó con el objetivo de verificar el cumplimiento de los atributos de calidad definidos, generados como consecuencia de la priorización de los requerimientos específicos de los atributos de calidad observados como escenarios. Teniendo en cuenta la priorización de cada escenario en base a dos dimensiones (D1, D2):

3.1.2. Dimensiones

D1: Importancia del escenario en relación al éxito del sistema, el cual emplea para su valoración los siguientes criterios:

Tabla 6 Criterios de valoración de la dimensión D1

Importancia	Descripción	Valor
Baja (B)	Importancia casi nula para el éxito del sistema	1
Media (M)	Importancia media para el éxito del sistema	2
Alta (A)	Importancia alta para el éxito del sistema	3

D2: Grado de dificultad que posee el escenario para ser realizado por la arquitectura, según la estimación del arquitecto:

Tabla 7 Criterios de valoración de la dimensión D2

Dificultad	Descripción	Valor
Bajo (B)	Esfuerzo bajo para cumplir el escenario	1
Medio (M)	Esfuerzo medio para cumplir el escenario	2
Alto (A)	Esfuerzo alto para cumplir el escenario	3

Tabla 8 Valoración del peso

Peso	Valor	Color
Baja	1 o 2	Verde
Media	3 o 4	Amarillo
Alta	Mayor a 6	Rojo

Tabla 9 Relación de Importancia y Esfuerzo

Importancia \ esfuerzo	Bajo	Medio	Alto
Baja	1	2	3
Media	2	4	6

Alta	3	6	9

3.1.3. Listado de escenarios

Se muestra el listado de escenarios, este listado de escenarios se obtuvo a partir de los criterios propuestos por Gartner para seleccionar los mejores CMT(Gartner 2018) en cuanto al despliegue de sistema operativo, la gestión de inventarios, la distribución de software, la gestión de parches, monitoreo de uso de software y control remoto de dispositivos, es decir para automatizar la administración del sistemas y las funciones de soporte. En función de las carencias del sistema actual, se diseñaron los escenarios siguientes:

Tabla 10 Listado de escenarios priorizados

No	Atributo / sub-característica	Escenario	Evaluación	Puntaje	Impacto (%)
1	Disponibilidad	Implantar la solución con soporte para clientes sin sistema operativo.	A, B	3,1	100
2	Disponibilidad	Si el sistema reporta mensajes de error, seguirá funcionando normalmente.	A, M	3,2	90
3	Disponibilidad	Si existe un fallo eléctrico, el sistema puede iniciar nuevamente sin problemas.	A, B	3,1	75
4	Disponibilidad	Se puede instalar el sistema en diferentes sistemas operativos basados en Linux.	A, B	3,1	100
5	Rendimiento	Se pueden hacer solicitudes simultáneas al cliente.	A, M	3,2	90

6	Seguridad	Se puede ver el contenido encriptado en los paquetes enviados hacia o desde el cliente.	A, A	3,3	100
***7**	Seguridad	Se puede comprobar que el envío de ficheros hacia el cliente es seguro usando sumas de verificación.	A, M	3,2	100
***8**	Seguridad	La información recopilada para el rendimiento se almacena en un repositorio seguro y con autenticación.	A, B	3,1	90
***9**	Modificabilidad	Se puede definir políticas informáticas generales, de clase o individuales para cada tipo de equipo.	A, A	3,3	100
10	Seguridad	Se almacena un historial (log) de las acciones realizadas en los equipos clientes y en el sistema.	A, B	3,1	75
***11**	Disponibilidad	Si ocurre algún problema y no se puede acceder al repositorio de información del sistema, este consta con una copia de seguridad, guardada en un repositorio de contingencia o localmente.	A, M	3,2	100

12	Seguridad	Solo el administrador de la red y otros usuarios con permisos de super-usuario podrán trabajar utilizando la herramienta.	A, M	3,2	95
13	Modificabilidad	Se utilizan archivos de configuración modificables y con un nivel fácil de comprensión de los parámetros.	A, B	3,1	90
14	Usabilidad	Se utiliza el modelo común de información (CIM) y aplicaciones que trae incorporada el sistema operativo.	A, B	3,1	95
15	Modificabilidad	De no estar incluida alguna configuración o paquete de aplicación en el cliente, se pueden enviar hacia el mismo.	A, M	3,2	100
16	Rendimiento	Los paquetes enviados a través de la red de trabajo, casi nunca exceden los 100 KB de información, excepto en los casos que sea necesario el envío de algún dato o fichero.	A, B	3,1	90
17	Disponibilidad	No se necesita interacción con el usuario para trabajar en los clientes.	A, M	3,2	100

18	Disponibilidad	Se ejecutan de forma remota todas las acciones en los clientes.	A, M	3,2	100
19	Rendimiento	Se pueden abortar las acciones, siempre y cuando no se hayan completado aún.	A, M	3,2	80

La fórmula utilizada para el cálculo de los pesos relativos finales es:

$$Pre_i = Pe_i / \sum_{i=0}^{n} Pe_i$$

Pre = peso relativo del escenario

Pe = peso del escenario

Atributo	Peso relativo del escenario (%)
Disponibilidad	95.90
Modificabilidad	86
Seguridad	95
Rendimiento	86
Usabilidad	95

Los cálculos realizados se encuentran el *Anexo 3*

Estos cálculos son el resultado de un promedio entre los pesos de los escenarios para cada atributo. Dicho peso se obtiene haciendo uso de las tablas de la relación entre importancia y esfuerzo y los campos de las columnas evaluación, puntaje e impacto de la tabla del listado de los escenarios priorizados.

Como se observa en la figura a continuación se puede hacer una comparación de los estados anteriores y después de los atributos de calidad extraídos para las arquitecturas de software. Se nota un aumento en los atributos principales de la investigación, modificabilidad, disponibilidad, seguridad y usabilidad. Solo se afecta el rendimiento, el cual se puede justificar, ya que se introducen nuevos recursos en la red y se proveen de métodos para aumentar los valores de los anteriores atributos.

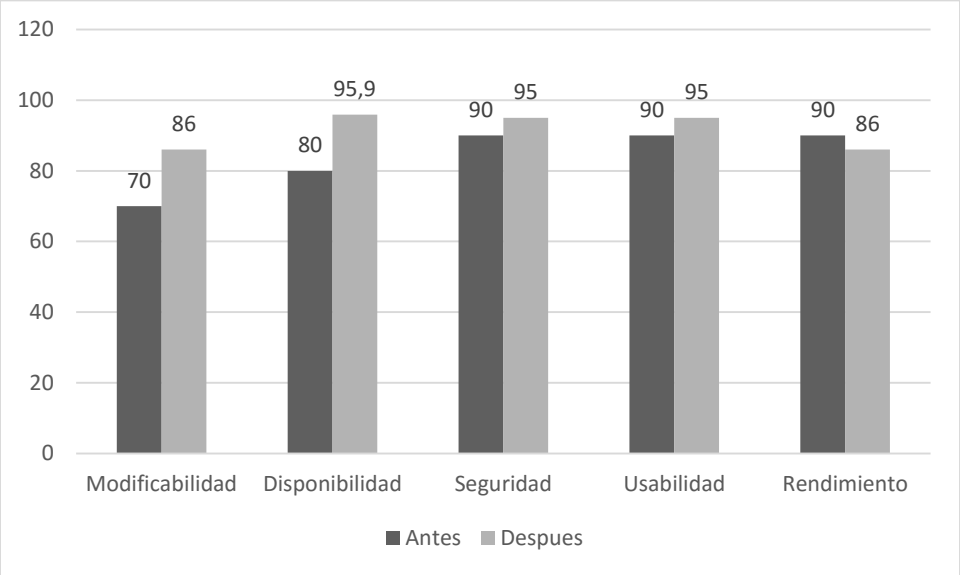

Figura 11 Comparación de los estados Anterior y Después de aplicar la solución.

3.2. Validación con software

Se realizó un software que implementa la arquitectura de la solución, con el cual apoyándose en los escenarios propuestos en ATAM, se procede a validar la misma. Se realiza una caracterización y vista de las respuestas de los escenarios implementados a continuación.

3.2.1. Escenario 1

Implantar la solución con soporte para clientes sin sistema operativo.

Este propone que se tenga soporte para clientes sin sistema operativo, utilizando protocolos o herramientas para la gestión de los mismos. Se emplea WoL, para despertar al ordenador clientes de su estado de apagado, suspensión o hibernación, entrando el mismo en estado activo y listo para ejecutar las órdenes del sistema gestor. Para la implementación del escenario se utiliza como estilo arquitectónico la gestión basada en estados, el cual propone solamente ejecutar a dispositivos con un cierto estado, características que solo sean aplicables a estos. En el caso particular de la herramienta implementada, se considera como estado, actividad o inactividad del cliente en la red, considerando Activo a los agentes que respondan a las peticiones del gestor e Inactivos a los que no den respuesta a las mismas. El envío del paquete solo se hace a dispositivos que posean el estado Inactivo. A continuación, se ve la captura del paquete WoL enviado a un dispositivo en estado Inactivo.

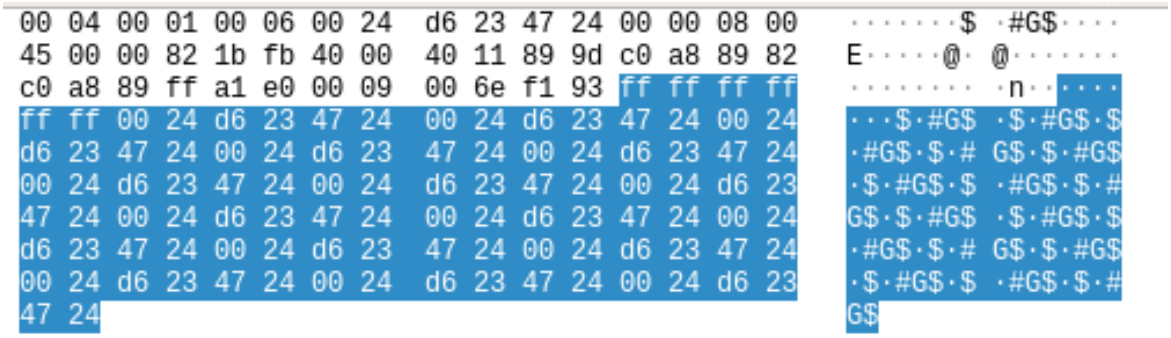

Figura 12 Captura del paquete WoL utilizando Wireshark.

3.2.2. Escenario 2

Si el sistema reporta mensajes de error, seguirá funcionando normalmente.

Este escenario se basa sobre la disponibilidad del sistema, en el cual, si son generados errores durante la ejecución del mismo, seguiría funcionando correctamente y haciendo uso del gestor de registros o del historial de fallos, notificar al administrador de lo ocurrido. Las decisiones arquitectónicas de este escenario, están enfocadas al trabajo en tiempo real conjuntamente con la gestión centralizada de errores y eventos que puedan ser generados de los mismos.

3.2.3. Escenario 4

Se puede instalar el sistema en diferentes sistemas operativos basados en Linux.

Tanto la aplicación gestora, como la agente, son desarrolladas utilizando el framework Qt sobre el lenguaje C++ en su versión C++17. Dicho framework se caracteriza por su amplia distribución en diversos sistemas operativos, dentro de los cuales se encuentran sistemas operativos Windows, MAC OS X, Linux y diferentes ramas de Unix. Haciendo uso de la portabilidad del framework y de la filosofía que sigue el mismo, se desarrollan las aplicaciones solo una vez y se compilan en cada uno de los sistemas en los que van a ser instaladas, respetando siempre las arquitecturas que posean los procesadores de los ordenadores, ya que pueden existir problemas de compatibilidad. Por lo que se tiene que generar un instalador para cada arquitectura en la que vaya a desplegar los agentes (dígase i386, amd64, entre otras).

3.2.4. Escenario 5

Se pueden hacer solicitudes simultáneas al cliente.

Si un cliente está siendo gestionado al mismo tiempo por varios equipos, esto no afecta su disponibilidad, debido a que su base de funcionamiento está sobre el trabajo con hilos. Si un proceso está utilizando algún recurso que no puede ser compartido, sería el único caso en que no pudieran

acceder los gestores al mismo recurso, siendo esto una limitación de los sistemas operativos instalados en las estaciones y no de la solución implementada.

3.2.5. Escenario 10

Se almacena un historial de las acciones realizadas en los equipos clientes y en el sistema.

Como el objetivo de mejorar la seguridad del sistema, se emplea un mecanismo para la gestión de las trazas en el cual se muestre información detallada de las acciones realizadas, categorizadas como "Informativas, Alertas y Críticas". Dichas acciones son almacenadas en un fichero de registro con variables de control de tiempo, usuario que ejecuta la acción y la acción ejecutada o error ocurrido.

```
void grhsserver::writeLog(QString txt)
{
    QString f = settings->value("logfile","history.log").toString();
    QString logFile = QDir::currentPath() + "/" + f;

    QFile file (logFile);
    QTextStream out(&file);

    if(!file.open(QIODevice::Append)){
        qDebug() << "No se pudo cargar el fichero " << file.fileName();
    }else{
        QString log = QString("%1-%2 --> Usuario: %3 --> %4\n").arg(QDate::currentDate().toString("dd.MM.yyyy"),
                                                                    QTime::currentTime().toString("hh:mm:ss"),
                                                                    QDir::homePath().section("/", -1),
                                                                    txt);

        out << log;
    }
}
```

Figura 13 Mecanismo de gestión de historial.

3.2.6. Escenario 12

Solo el administrador de la red y otros usuarios con permisos de super-usuario podrán trabajar utilizando la herramienta.

La herramienta solamente puede ser ejecutada por usuarios con permisos especiales. En el caso del agente, se ejecuta como un servicio, una vez arranque el sistema operativo y se ejecuta como usuario administrador (root). El gestor posee su mecanismo para detectar si el usuario que intenta ejecutar la misma, posee o no, los permisos necesarios para acceder a las funcionalidades de esta.

```
//check for sudo
auto me = getuid();

if (me){
    showMessageBox("Se necesita ejecutar como \'ROOT\'");
    exit(0);
}
```

Figura 14 Método de control de acceso en el gestor.

3.2.7. Escenario 13

Se utilizan archivos de configuración modificables y con un nivel fácil de comprensión de los parámetros.

Haciendo uso de las facilidades que proveen actualmente los frameworks de desarrollo de software con respecto a las configuraciones de los sistemas, se crean parámetros en la aplicación que pueden ser configurables o adaptables a las condiciones del usuario. Aumentando grandemente la modificabilidad que poseen las soluciones de gestión de clientes de red.

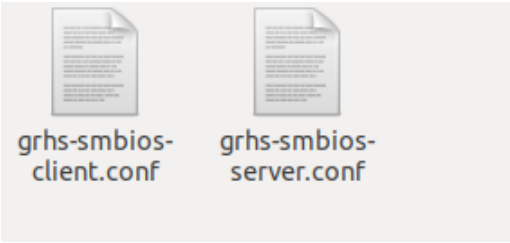

Figura 15 Ficheros de configuración de las soluciones Cliente y Servidor

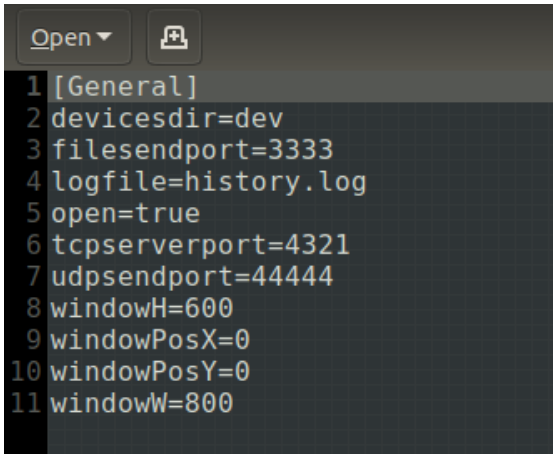

Figura 16 Parámetros ajustables en la solución Servidor

3.2.8. Escenario 16

Los paquetes enviados a través de la red de trabajo, casi nunca exceden los 100 KB de información, excepto en los casos que sea necesario el envío de algún fichero.

Durante el intercambio entre el cliente y el servidor, las comunicaciones más comunes de datos, oscilan entre los 4-100 bytes de información. Las solicitudes de información de los dispositivos

oscilan entre los 10-100 kb. Las ejecuciones de las políticas dependen del tamaño del script y la cantidad de dispositivos a los que sea enviado el mismo.

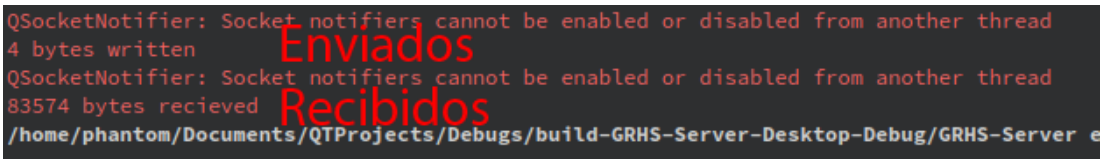

Figura 17 Tamaño de los datos de una solicitud de información del dispositivo.

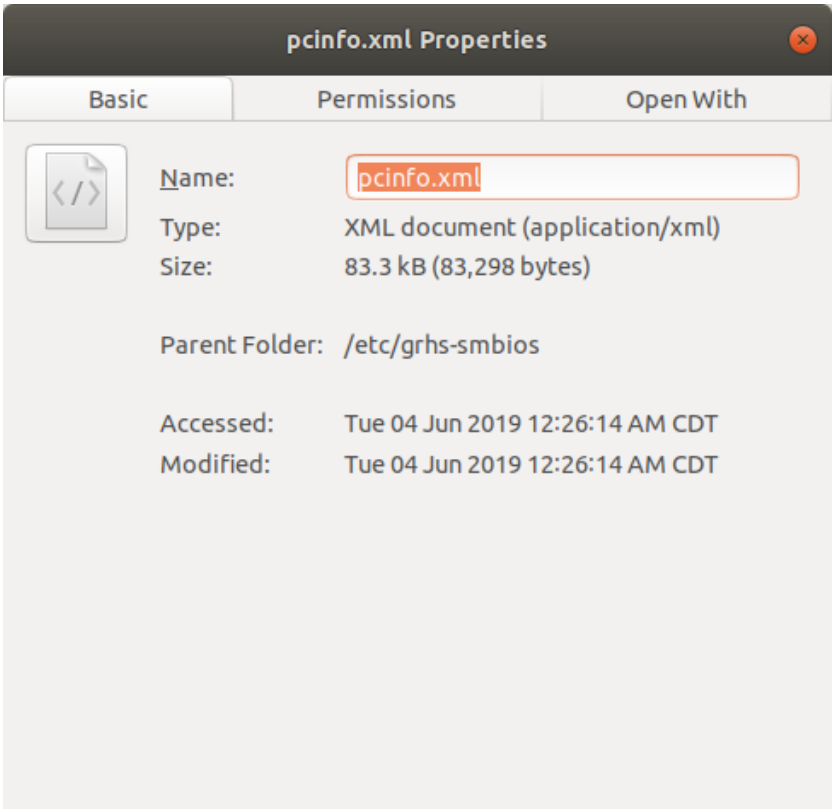

Figura 18 Tamaño promedio de un fichero de información del dispositivo.

3.2.9. Escenario 17

No se necesita interacción con el usuario para trabajar en los clientes.

Como se explicaba anteriormente en el epígrafe 3.3.6, la versión agente de la herramienta se ejecuta con permisos administrativos una vez que arranque el sistema operativo con permisos de administrador. Siendo un servicio que se ejecuta en segundo plano, es transparente al usuario en la estación de trabajo, en la mayoría de los casos que el equipo esté siendo controlado, el usuario no se percataría de lo que se puede estar realizando y no se necesita ningún tipo de aceptación por parte de estos.

3.2.10. Escenario 18

Se ejecutan de forma remota todas las acciones en los clientes.

Considerando que los clientes son máquinas físicas sobre las que pueden ejecutarse múltiples máquinas virtuales, las peticiones, eventos y acciones son realizadas de forma remota. Se posee un mecanismo de detección de errores en las solicitudes enviadas, en caso de que ocurra algún fallo en la realización de un comando, que se posean parámetros mal escritos o que no sean válidos.

3.3. Conclusiones del capítulo

A partir de los resultados obtenidos de la validación de la arquitectura propuesta a través del método ATAM y con la implementación de funcionalidades para un CMT de los diferentes escenarios, quedó demostrado la capacidad de alineamiento de la arquitectura de software para la gestión de clientes basado en políticas en entornos tecnológicos inestables a las necesidades de las organizaciones.

Conclusiones

Una vez diseñada la arquitectura de software para la gestión de clientes de red basado en políticas se verificó el alineamiento de esta en entornos tecnológicos inestables y se arriba a las conclusiones siguientes:

- El análisis de las arquitecturas de gestión de clientes de red permitió la identificación de las características que estas deben tener para su alineamiento en las necesidades de las organizaciones.
- El estudio las metodologías de diseño de arquitecturas de software permitió la selección de ABD aprovechando sus posibilidades para trabajar de manera integrada la combinación de requerimientos de negocio, de calidad y funcionales, garantizando la completitud de la herramienta desde su diseño.
- El empleo de gestión controlada por políticas, evita el aumento del tráfico en la red por el constante monitoreo que es necesario.
- El empleo de ATAM para validar la arquitectura diseñada en ABD se ve amplificado, ya que ABD tiene mini fases de ATAM dentro de sus etapas, permitiendo dentro de esto el ahorro de tiempo y recursos.

Recomendaciones

Teniendo en cuenta los resultados obtenidos en la realización del presente trabajo de diploma, se recomienda:

- Incorporar los resultados de la investigación realizada al Xilema-GRHS.
- Despliegue de la arquitectura de software en un entorno real realizando pruebas de rendimiento.
- Implementación de un sistema automatizado para la gestión de políticas TI incorporándole técnicas de inteligencia artificial.

Referencias Bibliográficas

BACHMANN, F., BASS, L., CHASTEK, G., DONOHOE, P. y PERUZZI, F., 2000. The Architecture Based Design Method, CMU/SEI-00-TR-001: CARNEGIE-MELLON UNIV PITTSBURGH PA SOFTWARE ENGINEERING INSTITUTE.

BARÓ MENÉNDEZ, D., ANIAS CALDERÓN, C. y PEÑA CASANOVA, M., 2018. Propuesta de arquitectura para la PBNM utilizando WBEM y herramientas de software libre y código abierto. I Conferencia Científica Internacional UCIENCIA 2014, Universidad de las Ciencias Informáticas. La Habana, Cuba

CASANOVA, M.P. y CALDERÓN, C.A., 2018. Selección de modelos de información para gestión integrada de redes y servicios basada en políticas. *Revista Científica de Ingeniería Electrónica, Automática y Comunicaciones ISSN: 1815-5928*, vol. 39, no. 3, pp. 77-88. ISSN 1815-5928.

CAVALCANTE, F., JUCA, P., ALENCAR, J.M. y CALLADO, A., 2018. Remote Access with Wake on LAN. *EATIS '18*. S.l.: s.n., pp. 8. ISBN 978-1 -4503-6572-7/18/11. DOI https://doi.org/10.1145/3293614.3293621.

CHACON, S. y STRAUB, B., 2014. *Pro Git*. Apress 2da. Edición, Libro Electrónico, Berkeley, CA.

CHRISTENSSON, 2016. NMS (Network Management System) Definition. [en línea]. [Consulta: 26 febrero 2019]. Disponible en: https://techterms.com/definition/nms.

COMPANY, T.Q., 2016. Legal | Licensing - Qt. [en línea]. [Consulta: 21 mayo 2019]. Disponible en: https://www.qt.io/licensing/.

FIELDING, R.T., 2000. *Architectural Styles and the Design of Network-based Software Architectures*. Tesis Doctoral. Irvine: University of California.

GARTNER, 2017. IT Services - Gartner IT Glossary. [en línea]. [Consulta: 23 abril 2019]. Disponible en: https://www.gartner.com/it-glossary/it-services.

GARTNER, 2018. Client Management Tools - Gartner IT Glossary. [en línea]. [Consulta: 7 febrero 2019]. Disponible en: https://www.gartner.com/it-glossary/client-management-tools.

GITLAB, 2019a. GitLab Contributors - All time. [en línea]. [Consulta: 21 mayo 2019]. Disponible en: http://contributors.gitlab.com/contributors.

GITLAB, 2019b. Team. *GitLab* [en línea]. [Consulta: 21 mayo 2019]. Disponible en: https://about.gitlab.com/company/team/.

HPE, 2019. What is a Reference Architecture? - Enterprise IT Definitions | HPE. [en línea]. [Consulta: 19 marzo 2019]. Disponible en: https://www.hpe.com/us/en/what-is/reference-architecture.html.

INTEL CORPORATION y SYSTEMSOFT, 1999. *Preboot Execution Environment (PXE) Specification*. 20 septiembre 1999. S.l.: s.n.

KARUNA JYOTHI, K. y REDDY, B.I., 2018. Study on Virtual Private Network (VPN), VPN's Protocols And Security. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 3, no. 5, pp. 14. ISSN 2456-3307.

KAZMAN, R., KLEIN, M. y CLEMENTS, P., 2000. *ATAM: Method for Architecture Evaluation*. 2000. S.l.: s.n.

KLEIN, M., KAZMAN, R., BASS, L., CARRIERE, BARBACCI y LIPSON, 1999. Attribute Based Architectural Styles. *Proceedings of the 1st Working IFIP Conferenceon Software Architecture (WICSA1)*. San Antonio, TX: s.n.,

KOSIUR, D., 2001. *Understanding Policy-Based Networking*. S.l.: John Wiley & Sons. ISBN 978-0-471-01374-7.

KRUCHTEN, P., 1995. *The 4+1 View Model of Architecture*. 6th Edition. , **ISSN:** 0740-7459, IEEE Software, **DOI:** 10.1109/52.469759

LARMAN, C., 2004. *UML y Patrones*. 2da. S.l.: PEARSON EDUCACION.

MILLÁN TEJEDOR, R.J., 1999. Gestión de red. *Windows NT/2000*, no. 12.

PASTOR FRANCO, J.A., 2002. *Evaluación y desarrollo incremental de una arquitectura software de referencia para sistemas de teleoperación utilizando métodos formales*. Tesis Doctoral. Dpto. de Tecnologías de la Información y de la Comunicación: Universidad Politécnica de Cartagena.

QT FOUNDATION, 2018. Qt Creator: FAQ — Qt Software - Code Less. Create More. Deploy Everywhere. [en línea]. [Consulta: 21 mayo 2019]. Disponible en: https://web.archive.org/web/20081222010435/http://trolltech.com/developer/qt-creator/faq#on-which-operating-systems.

ROUSE, M., 2006. What is network interface card (NIC)? - Definition from WhatIs.com. *SearchNetworking* [en línea]. [Consulta: 31 enero 2019]. Disponible en: https://searchnetworking.techtarget.com/definition/network-interface-card.

ROUSE, M., 2007. What is Web services (application services)? - Definition from WhatIs.com. *SearchMicroservices* [en línea]. [Consulta: 23 abril 2019]. Disponible en: https://searchmicroservices.techtarget.com/definition/Web-services-application-services.

ROUSE, M., 2016. What is operating system (OS)? - Definition from WhatIs.com. *WhatIs.com* [en línea]. [Consulta: 15 enero 2019]. Disponible en: https://whatis.techtarget.com/definition/operating-system-OS.

RUMBAUGH, JACOBSON y BOOCH, 2007. *El lenguaje unificado de modelado*. 2nd. Madrid: Addison Wesley. ISBN 978-84-7829-087-1.

STRASSNER, J., 2004. Chapter 7 - The DEN-ng Policy Model. *Policy-Based Network Management* Burlington: Morgan Kaufmann, The Morgan Kaufmann Series in Networking, pp. 259-308. [Consulta: 28 febrero 2018]. Disponible en: https://www.sciencedirect.com/science/article/pii/B9781558608597500402.

TRAVER SALCEDO, 2002. *Propuesta de Arquitectura de Referencia de Sistemas de e-Salud y e-Inclusión*. Tesis Doctoral. S.l.: Universidad Politécnica de Valencia.

VISUAL PARADIGM, 2015. Ideal Modeling & Diagramming Tool for Agile Team Collaboration. [en línea]. [Consulta: 21 mayo 2019]. Disponible en: https://www.visual-paradigm.com/.

Bibliografía

ACEBO PLAZA, M., NÚÑEZ, A., VILLAVICENCIO, M., RODRÍGUEZ, J., ZAMBRANO, J. y QUIJANO, J., 2017. ESTUDIOS INDUSTRIALES ORIENTACIÓN ESTRATÉGICA PARA LA TOMA DE DECISIONES.

ASENSIO-PÉREZ, J., VILLAGRA, V., LÓPEZ DE VERGARA MÉNDEZ, J. y BERROCAL, J., 1999. Experiences with the SNMP-based integrated management of a CORBA-based electronic commerce application. S.l.: s.n., pp. 517-530. DOI 10.1109/INM.1999.770705.

BACHMANN, F., BASS, L., CHASTEK, G., DONOHOE, P. y PERUZZI, F., 2000. The Architecture Based Design Method. S.l.:

BARO MENÉNDEZ, D., 2015. *Arquitectura de Referencia para una Plataforma Integral de Telecomunicaciones Unificadas, ARPlatTel-U*. Tesis Maestría. La Habana, Cuba: Universidad de las Ciencias Informáticas.

BARÓ MENÉNDEZ, D., ANIAS CALDERÓN, C. y PEÑA CASANOVA, M., 2018. *Propuesta de arquitectura para la PBNM utilizando WBEM y herramientas de software libre y código abierto*. S.l.: s.n.

BASS, L., CLEMENTS, P. y KAZMAN, R., 2012. *Software Architecture in Practice: Software Architect Practice_c3*. S.l.: Addison-Wesley. ISBN 978-0-13-294278-2.

CAVALCANTE, F., JUCA, P., ALENCAR, J.M. y CALLADO, A., 2018. Remote Access with Wake on LAN. *EATIS '18*. S.l.: s.n., pp. 8. ISBN 978-1 -4503-6572-7/18/11. DOI https://doi.org/10.1145/3293614.3293621.

CHACON, S. y STRAUB, B., 2014. *Pro Git*. 2da. S.l.: Apress.

CHRISTENSSON, 2016a. NMS (Network Management System) Definition. [en línea]. [Consulta: 26 febrero 2019]. Disponible en: https://techterms.com/definition/nms.

CHRISTENSSON, 2016b. Operating System Definition. [en línea]. [Consulta: 26 febrero 2019]. Disponible en: https://techterms.com/definition/operating_system.

CHRISTENSSON, 2018. NIC (Network Interface Card) Definition. [en línea]. [Consulta: 26 febrero 2019]. Disponible en: https://techterms.com/definition/nic.

COMPANY, T.Q., 2016. Legal | Licensing - Qt. [en línea]. [Consulta: 21 mayo 2019]. Disponible en: https://www.qt.io/licensing/.

CONNERY, G.W. y NESSETT, D.M., 2003. Secure system for remote management and wake-up commands. US6606709B1. US6606709B1. US10075103

COURTEMACHE, M., 2015. Why should I use a configuration management system? *Search IT Operations* [en línea]. [Consulta: 3 octubre 2018]. Disponible en: https://searchitoperations.techtarget.com/answer/Why-should-I-use-a-configuration-management-system.

DÍAZ, W.M.F., 2010. *Definición de un modelo genérico para la caracterización de escenarios virtuales de redes IP*. http://purl.org/dc/dcmitype/Text. S.l.: Universidad Autónoma de Madrid.

DING, J., 2016. *Advances in Network Management*. S.l.: CRC Press. ISBN 978-1-4200-6455-1.

DORDOIGNE, J., 2018. *Redes Informáticas - Nociones fundamentales*. 6. S.l.: Ediciones ENI. ISBN 978-2-409-01279-2.

FIELDING, R.T., 2000. *Architectural Styles and the Design of Network-based Software Architectures*. Tesis Doctoral. Irvine: University of California.

GARRETT, H.M., JR, F.E.N. y TOMEK, L.A., 2000. Wake multiple over LAN. US6047378A. US6047378A. US08940106

GARTNER, 2017. IT Services - Gartner IT Glossary. [en línea]. [Consulta: 23 abril 2019]. Disponible en: https://www.gartner.com/it-glossary/it-services.

GARTNER, 2018. Client Management Tools - Gartner IT Glossary. [en línea]. [Consulta: 7 febrero 2019]. Disponible en: https://www.gartner.com/it-glossary/client-management-tools.

GITLAB, 2019a. GitLab Contributors - All time. [en línea]. [Consulta: 21 mayo 2019]. Disponible en: http://contributors.gitlab.com/contributors.

GITLAB, 2019b. Team. *GitLab* [en línea]. [Consulta: 21 mayo 2019]. Disponible en: https://about.gitlab.com/company/team/.

GUERRERO-CASTELEIRO, A., 2007. *Especificación del comportamiento de gestión de red mediante ontologías*. Tesis Doctoral. S.l.: E.T.S.I. Telecomunicación (UPM).

HPE, 2019. What is a Reference Architecture? - Enterprise IT Definitions | HPE. [en línea]. [Consulta: 19 marzo 2019]. Disponible en: https://www.hpe.com/us/en/what-is/reference-architecture.html.

IEEE ARCHITECTURE WORKING GROUP, 2000. IEEE Standard 1471-2000, Recommended practice for Architectural Description of Software-Intensive Systems. *IEEE*, ISSN 978-0-7381-2519-0. DOI https://doi.org/10.1109/IEEESTD.2000.91944.

IN, H., KAZMAN, R. y OLSON, D., 2001. From requirements negotiation to software architectural decisions. S.l.:

INTEL, 2019. Intel® AMT and the Intel® ME. [en línea]. Disponible en: https://software.intel.com/en-us/blogs/2011/12/14/intelr-amt-and-the-intelr-me.

INTEL CORPORATION y SYSTEMSOFT, 1999. *Preboot Execution Environment (PXE) Specification*. 20 septiembre 1999. S.l.: s.n.

KARUNA JYOTHI, K. y REDDY, B.I., 2018. Study on Virtual Private Network (VPN), VPN's Protocols and Security. *International Journal of Scientific Research in Computer Science, Engineering and Information Technology*, vol. 3, no. 5, pp. 14. ISSN 2456-3307.

KAZMAN, R., KLEIN, M. y CLEMENTS, P., 2000. *ATAM: Method for Architecture Evaluation*. 2000. S.l.: s.n.

KLEIN, M., KAZMAN, R., BASS, L., CARRIERE, BARBACCI y LIPSON, 1999. Attribute Based Architectural Styles. *Proceedings of the 1st Working IFIP Conference on Software Architecture (WICSA1)*. San Antonio, TX: s.n.,

KOSIUR, D., 2001. *Understanding Policy-Based Networking*. S.l.: John Wiley & Sons. ISBN 978-0-471-01374-7.

KRUCHTEN, P., 1995. *The 4+1 View Model of Architecture*. 6th. S.l.: IEEE Software.

LARMAN, C., 2004. *UML y Patrones*. 2da. S.l.: PEARSON EDUCACION.

LAZAR, A. y SARACCO, R., 2013. *Integrated Network Management V: Integrated management in a virtual world Proceedings of the Fifth IFIP/IEEE International Symposium on Integrated Network Management San Diego, California, U.S.A., May 12–16, 1997*. S.l.: Springer. ISBN 978-0-387-35180-3.

LEE, K.-W. y LOBO, J., **2015**. Policy-based management of networked computing systems. *ResearchGate*,

LEÓN, R.A.H. y COELLO GONZÁLEZ, S., 2014. *El proceso de investigación científica*. 2. S.l.: Editorial Universitaria. ISBN 978-959- 16-1557-2.

LOPEZ, D., LÓPEZ DE VERGARA MÉNDEZ, J., VIDAL, F. y SANCHEZ-MACIAN, A., 2018. An OWL-S based architecture for self-optimizing multimedia over ip services.

LÓPEZ DE VERGARA MÉNDEZ, J., VILLAGRA, V. y BERROCAL, J., 2018. Xml-based network management - Applying the web ontology language to management information definitions.

MILLÁN TEJEDOR, R.J., 1999. Gestión de red. *Windows NT/2000*, no. 12.

MOINDZE, S.M. y KONATE, K., 2014. A survey of the distributed network management models and architectures: Assessment and challenges. *2014 IEEE 6th International Conference on Adaptive Science Technology (ICAST)*. S.l.: s.n., pp. 1-8. DOI 10.1109/ICASTECH.2014.7068135.

PASTOR FRANCO, J.A., 2002. *Evaluación y desarrollo incremental de una arquitectura software de referencia para sistemas de teleoperación utilizando métodos formales*. Tesis Doctoral. Dpto. de Tecnologías de la Información y de la Comunicación: Universidad Politécnica de Cartagena.

PEISO CRUZ, M.E., 2018. *Procedimiento de construcción de imágenes de la personalización de GNU/Linux Nova Escritorio.* Trabajo final presentado en opción al título de Máster en Informática Avanzada. La Habana, Cuba: Universidad de las Ciencias Informáticas.

POSLAD, S., 2011. *Ubiquitous Computing: Smart Devices, Environments and Interactions*. S.l.: John Wiley & Sons. ISBN 978-1-119-96526-8.

PRESSMAN, R.S., 2010. *Ingeniería de software. Un enfoque práctico*. 7. University of Connecticut: Mc Graw Hill. ISBN 978-607-15-0314-5.

QT FOUNDATION, 2018. Qt Creator: FAQ — Qt Software - Code Less. Create More. Deploy Everywhere. [en línea]. [Consulta: 21 mayo 2019]. Disponible en: https://web.archive.org/web/20081222010435/http://trolltech.com/developer/qt-creator/faq#on-which-operating-systems.

REYNOSO, C.B., 2004. Introducción a la arquitectura de software.

ROUSE, M., 2006. What is network interface card (NIC)? - Definition from WhatIs.com. *SearchNetworking* [en línea]. [Consulta: 31 enero 2019]. Disponible en: https://searchnetworking.techtarget.com/definition/network-interface-card.

ROUSE, M., 2007. What are Web services (application services)? - Definition from WhatIs.com. *Search Microservices* [en línea]. [Consulta: 23 abril 2019]. Disponible en: https://searchmicroservices.techtarget.com/definition/Web-services-application-services.

ROUSE, M., 2016. What is operating system (OS)? - Definition from WhatIs.com. *WhatIs.com* [en línea]. [Consulta: 15 enero 2019]. Disponible en: https://whatis.techtarget.com/definition/operating-system-OS.

RUMBAUGH, JACOBSON y BOOCH, 2007. *El lenguaje unificado de modelado*. 2nd. Madrid: Addison Wesley. ISBN 978-84-7829-087-1.

RYU, C.-H., 2003. Method and apparatus for controlling power of computer system using wake up LAN (local area network) signal. US6591368B1. US6591368B1. US09/363,658

SCHOENWAELDER, J., HARRINGTON, D., WEISS, W., JASON, J., STRAUSS, F. y ELLIOTT, C., 2018. SMIng Objectives.

SOMMERVILLE, I., 2011. *Ingeniería de Software*. México: PEARSON EDUCACIÓN. ISBN 978-607-32-0603-.

STRASSNER, J., 2003. *Policy-Based Network Management: Solutions for the Next Generation (The Morgan Kaufmann Series in Networking)*. San Francisco, CA, USA: Morgan Kaufmann Publishers Inc. ISBN 978-1-55860-859-7.

STRAUSS, F. y SCHOENWAELDER, J., 2018. SMIng - Next Generation Structure of Management Information.

SUGANAMI, K., 2011. MAC filtering on ethernet PHY for wake-on-LAN. US8799633B2. US8799633B2.

TRAVER SALCEDO, 2002. *Propuesta de Arquitectura de Referencia de Sistemas de e-Salud y e-Inclusión*. Tesis Doctoral. S.l.: Universidad Politécnica de Valencia.

TSAROUCHIS, C., DENAZIS, S., KITAHARA, C., VIVERO, J., SALAMANCA, E., MAGANA, E., GALIS, A., MANAS, J.L., CARLINET, L., MATHIEU, B. y KOUFOPAVLOU, O., 2003. A policy-based management

architecture for active and programmable networks. *IEEE Network*, vol. 17, no. 3, pp. 22-28. ISSN 0890-8044. DOI 10.1109/MNET.2003.1201473.

UCI, 2016. Misión | Universidad de las Ciencias Informáticas. [en línea]. [Consulta: 14 mayo 2019]. Disponible en: http://www.uci.cu/universidad/mision.

VISUAL PARADIGM, 2015. Ideal Modeling & Diagramming Tool for Agile Team Collaboration. [en línea]. [Consulta: 21 mayo 2019]. Disponible en: https://www.visual-paradigm.com/.

VOLPENTESTA, J.R., 2016. El impacto de las TIC sobre las estructuras organizacionales y el trabajo del hombre en las empresas. *FACES*, vol. 22, no. 46, pp. 81-94. ISSN 0328-4050.

WESTERINEN, A., SCHNIZLEIN, J., STRASSNER, J., SCHERLING, M., QUINN, B., HERZOG, S., HUYNH, A., CARLSON, M., PERRY, J. y WALDBUSSER, S., 2001. Terminology for Policy-Based Management, ISSN 2070-1721.

WOJCIK, BACHMANN, F., BASS, L., CLEMENTS, P., MERSON, NORD y WOOD, 2006. Attribute-driven design (add). Informe técnico. S.l.: Carnegie Mellon University.

YOUNG, J. y COPELAND, B.W., 2005. Method and system for remote client installation. US6966060B1. US6966060B1. US09/599,156

Anexo 1 Encuesta a expertos

Encuesta realizada a los expertos del sistema XILEMA-GRHS en el Centro de Telemática en la UCI.

Estimado experto, la siguiente encuesta ha sido elaborada para identificar los requisitos a considerar en el diseño de una **Arquitectura de Gestión Basada en Políticas TI**. Por tal motivo se le solicita que considere los requisitos de la misma a partir de las cuestiones que se enuncian a continuación:

Gracias de antemano por su cooperación.

1. Es necesario realizar la gestión de máquinas clientes o dispositivos móviles en ausencia de sistema operativo o cuando se encuentran apagados.				
Muy de acuerdo	De acuerdo	Ni de acuerdo ni en desacuerdo	En desacuerdo	Muy en desacuerdo
2. Es importante que las herramientas de gestión que se desarrollen en la universidad sean de código abierto.				
Muy de acuerdo	De acuerdo	Ni de acuerdo ni en desacuerdo	En desacuerdo	Muy en desacuerdo
3. Es importante lograr que la Arquitectura de Gestión Basada en Políticas TI garantice una alta capacidad de integración entre las herramientas de gestión que formen parte de ella.				
Muy de acuerdo	De acuerdo	Ni de acuerdo ni en desacuerdo	En desacuerdo	Muy en desacuerdo
4. Es importante garantizar en Arquitectura de Gestión Basada en Políticas TI la capacidad de integrar nuevas clases de entidades de infraestructura TI en el entorno de gestión.				
Muy de acuerdo	De acuerdo	Ni de acuerdo ni en desacuerdo	En desacuerdo	Muy en desacuerdo

5. Es importante que la Arquitectura de Gestión Basada en Políticas TI permita identificar y solucionar conflictos entre políticas, facilite la automatización y contribuya a reducir la complejidad en la gestión.				
Muy de acuerdo	De acuerdo	Ni de acuerdo ni en desacuerdo	En desacuerdo	Muy en desacuerdo
Los aspectos considerados con anterioridad para el diseño de una Arquitectura de Gestión Basada en Políticas TI los considera:				
a) Bien concebidas	b) Haría cambios	c) Haría adiciones	d) Haría supresiones	
Siempre que Usted marque una de las columnas b), c) o d) especifique el cambio, adición o supresión que haría:				

Anexo 2 Escenarios del árbol de utilidad de ATAM

Escenarios del método ATAM

Escenario 1: Implantar la solución con soporte para clientes sin sistema operativo.

- **Atributo**: disponibilidad
- **Entorno**: ejecución normal
- **Estímulo**: envío de un paquete Wake on LAN
- **Respuesta**: enciende el cliente
- **Decisión Arquitectónica**:
    - Arquitectura basada en estados

Escenario 2: Si el sistema reporta mensajes de error, seguirá funcionando normalmente.

- **Atributo**: disponibilidad
- **Entorno**: ejecución normal
- **Estímulo**: ocurre un error durante la ejecución
- **Respuesta**: se informa del error y se almacena en un historial de errores
- **Decisión Arquitectónica**:
    - Arquitectura con un mecanismo centralizado de gestión de errores e historial
    - Arquitectura basada en eventos
    - Arquitectura de tiempo real

Escenario 3: Si existe un fallo eléctrico, el sistema puede iniciar nuevamente sin problemas.

- **Atributo**: disponibilidad
- **Entorno**: ejecución normal
- **Estímulo**: ocurre un fallo eléctrico durante la ejecución
- **Respuesta**: luego de encendido el ordenador servidor, se vuelve a ejecutar el sistema.
- **Decisión Arquitectónica**:
    - Basada en la puesta en marcha por ficheros de configuración

Escenario 4: Se puede instalar el sistema en diferentes sistemas operativos basados en Linux.

- **Atributo**: disponibilidad
- **Entorno**: momento de instalación
- **Estímulo**: al instalar la aplicación en un sistema operativo basado en Linux
- **Respuesta**: se instala normalmente y se configura los paquetes necesarios
- **Decisión Arquitectónica**:

Escenario 5: Se pueden hacer solicitudes simultáneas al cliente.

- **Atributo**: rendimiento
- **Entorno**: ejecución normal
- **Estímulo**: solicitud de una acción de forma remota
- **Respuesta**: se ejecuta dicha acción, si no está siendo bloqueada por el mismo sistema
- **Decisión Arquitectónica**:

Escenario 6: Se puede ver el contenido encriptado de los paquetes enviados hacia o desde el cliente.

- **Atributo**: rendimiento
- **Entorno**: ejecución normal
- **Estímulo**: lectura usando un "*sniffer*" de un paquete enviado por la aplicación
- **Respuesta**: se ve el texto plano de la codificación de la acción enviada
- **Decisión Arquitectónica**:

Escenario 7: Se puede comprobar que el envío de ficheros hacia el cliente es seguro usando sumas de verificación.

- **Atributo**: seguridad
- **Entorno**: ejecución normal
- **Estímulo**: envío de un fichero hacia el cliente o viceversa
- **Respuesta**: se envía una suma de verificación junto con el fichero enviado y se comprueba que el recibido sea el mismo, de no ser hacer se descarta.
- **Decisión Arquitectónica**:
    - Implementación de firmas digitales
    - Uso de protocoles de transmisión seguros

Escenario 8: La información recopilada para el rendimiento se almacena en un repositorio seguro y con autenticación.

- **Atributo**: seguridad
- **Entorno**: ejecución normal
- **Estímulo**: solicitud de información del repositorio
- **Respuesta**: se requiere de autenticación para acceder a la información
- **Decisión Arquitectónica**:
    - Arquitectura de repositorios

Escenario 9: Se puede definir políticas informáticas generales, de clase o individuales para cada tipo de equipo.

- **Atributo**: modificabilidad
- **Entorno**: elaboración de políticas informáticas
- **Estímulo**: se crea una política nueva
- **Respuesta**:
- **Decisión Arquitectónica**:
    - Patrón de implementación de políticas basado en GRASP
    - Control de acceso basado en roles

Escenario 10: Se almacena un historial de las acciones realizadas en los equipos clientes y en el sistema.

- **Atributo**: seguridad
- **Entorno**: ejecución normal
- **Estímulo**: se realiza una acción en el sistema u ocurre un error (evento)
- **Respuesta**: se registra en un archivo historial el evento ocurrido
- **Decisión Arquitectónica**:
    - Mecanismo centralizado de gestión de trazas

Escenario 11: Si ocurre algún problema y no se puede acceder al repositorio de información del sistema, este consta con una copia de seguridad, guardada en un repositorio de contingencia o localmente.

- **Atributo**: disponibilidad
- **Entorno**: ejecución normal
- **Estímulo**: no se puede acceder al repositorio
- **Respuesta**: el sistema lee de una copia de respaldo hecha anteriormente en otro servidor de contingencia o local
- **Decisión Arquitectónica**:
    - Backup server

Escenario 12: Solo el administrador de la red y otros usuarios con permisos de super-usuario podrán trabajar utilizando la herramienta.

- **Atributo**: seguridad
- **Entorno**: ejecución normal
- **Estímulo**: solicitud de una acción por el sistema hacia un cliente

- **Respuesta**: el sistema operativo verifica los permisos de usuario y se encarga de permitir o no la ejecución de la acción
- **Decisión Arquitectónica**:
    - Control de acceso basado en roles

Escenario 13: Se utilizan archivos de configuración modificables y con un nivel fácil de comprensión de los parámetros.

- **Atributo**: modificabilidad
- **Entorno**: configuración o instalación de cliente o servidor
- **Estímulo**: se quiere modificar parámetros en las aplicaciones
- **Respuesta**:
- **Decisión Arquitectónica**:

Escenario 14: Se utiliza el modelo común de información (CIM) y aplicaciones que vienen incorporadas en el sistema operativo.

- **Atributo**: usabilidad
- **Entorno**: desarrollo
- **Estímulo**: al utilizar datos que usen CIM
- **Respuesta**: se organizan, modelan, formulan, los datos en la forma que este propone
- **Decisión Arquitectónica**:
    - Encapsulamiento basado en el modelo común de información (CIM / XML)

Escenario 15: De no estar incluida alguna configuración o paquete de aplicación en el cliente, se pueden enviar hacia el mismo.

- **Atributo**: modificabilidad
- **Entorno**: ejecución normal
- **Estímulo**: se quiere ejecutar o instalar algún paquete que no se encuentra en el destino
- **Respuesta**: se recibe el paquete y se realiza la acción deseada
- **Decisión Arquitectónica**:

Escenario 16: Los paquetes enviados a través de la red de trabajo, casi nunca exceden los 100 KB de información, excepto en los casos que sea necesario el envío de algún fichero.

- **Atributo**: rendimiento
- **Entorno**: ejecución normal y envío de información
- **Estímulo**: se envía información del cliente al servidor o viceversa

- **Respuesta**: dicha información no satura la red, ya que es de poco tamaño
- **Decisión Arquitectónica**:

Escenario 17: No se necesita interacción con el usuario para trabajar en los clientes.

- **Atributo**: disponibilidad
- **Entorno**: ejecución normal
- **Estímulo**: se inicia el sistema operativo
- **Respuesta**: se ejecutan normalmente los comandos sin necesidad de que el usuario se encuentre presente
- **Decisión Arquitectónica**:
    - Modelo de diseño de tablas de decisión

Escenario 18: Se ejecutan de forma remota todas las acciones en los clientes.

- **Atributo**: disponibilidad
- **Entorno**: ejecución normal
- **Estímulo**: se realiza una acción
- **Respuesta**: se ejecuta la acción o devuelve un error
- **Decisión Arquitectónica**:

Escenario 19: Se pueden abortar las acciones, siempre y cuando no se hayan completado aún.

- **Atributo**: rendimiento
- **Entorno**: ejecución normal
- **Estímulo**: se realiza una acción
- **Respuesta**: se ejecuta la acción o devuelve un error
- **Decisión Arquitectónica**:

Anexo 3 Cálculos de los atributos de calidad

A continuación, se calcula los pesos relativos para cada atributo de calidad escogido.

Tabla 11 Cálculo de Peso Relativo del Escenario para Disponibilidad

Atributo	Puntaje	Impacto	Valor	E = Valor/Valor total	Pe = Impacto * E
Disponibilidad	**3,1**	100	3	0.090909091	9.090909091
Disponibilidad	**3,2**	90	6	0.181818182	16.36363636
Disponibilidad	**3,1**	75	3	0.090909091	6.818181818
Disponibilidad	**3,1**	100	3	0.090909091	9.090909091
Disponibilidad	**3,2**	100	6	0.181818182	18.18181818
Disponibilidad	**3,2**	100	6	0.181818182	18.18181818
Disponibilidad	**3,2**	100	6	0.181818182	18.18181818
Total			33		**95.90909091**

Tabla 12 Cálculo de Peso Relativo del Escenario para Modificabilidad

Atributo	Puntaje	Impacto	Valor	E = Valor/Valor total	Pe = Impacto * E
Modificabilidad	**3,1**	90	3	0.166666667	15
Modificabilidad	**3,2**	100	6	0.333333333	33.33333333
Modificabilidad	**3,3**	100	9	0.5	50
Total			18		**86**

Tabla 13 Cálculo de Peso Relativo del Escenario para Rendimiento

Atributo	Puntaje	Impacto	Valor	E = Valor/Valor total	Pe = Impacto * E
Rendimiento	**3,2**	80	6	0.4	32
Rendimiento	**3,1**	90	3	0.2	18
Rendimiento	**3,2**	90	6	0.4	36
Total			15		**86**

Tabla 14 Cálculo de Peso Relativo del Escenario para Seguridad

Atributo	Puntaje	Impacto	Valor	E = Valor/Valor total	Pe = Impacto * E
Seguridad	**3,3**	100	9	0.333333333	33.33333333
Seguridad	**3,2**	100	6	0.222222222	22.22222222
Seguridad	**3,1**	90	3	0.111111111	10
Seguridad	**3,1**	75	3	0.111111111	8.333333333
Seguridad	**3,2**	95	6	0.222222222	21.11111111
Total			27		**95**

Tabla 15 Cálculo de Peso Relativo del Escenario para Usabilidad

Atributo	Puntaje	Impacto	Valor	E = Valor/Valor total	Pe = Impacto * E
Usabilidad	**3,1**	95	3	1	95
Total			3		**95**

Anexo 4 Gasto en infraestructura TI

Gasto total en infraestructura de tecnología de la información tradicional de la empresa en hardware, software, servicio y personal en todo el mundo desde 2016 hasta 2027 (en miles de millones de dólares estadounidenses)

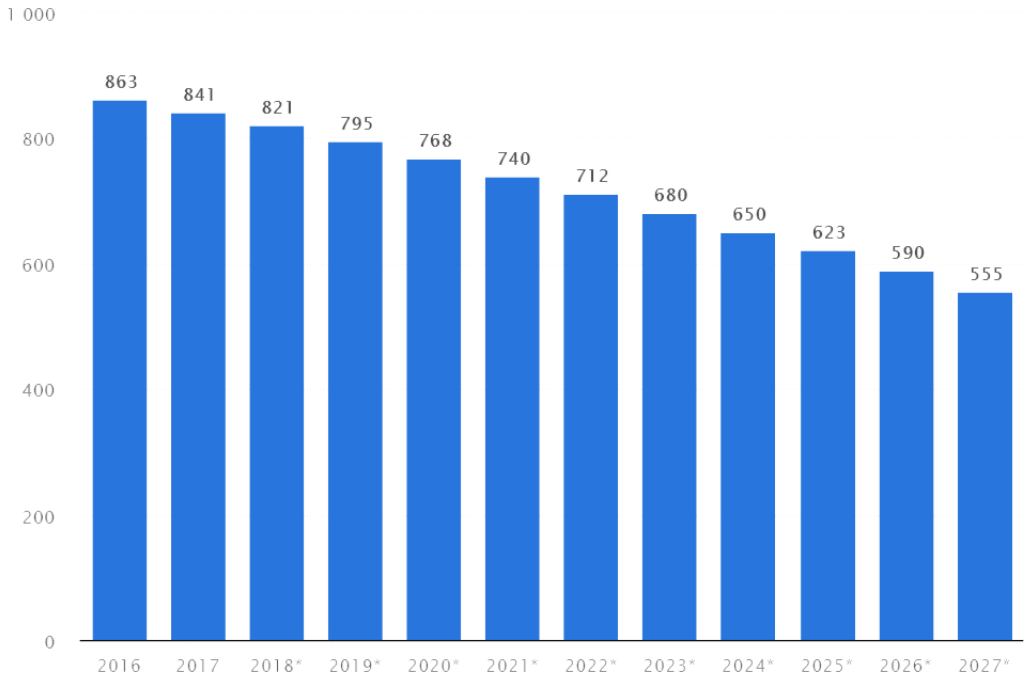

Figura 19 Gasto en miles de millones de dólares (USD)

Anexo 5 Pronóstico del gasto en hardware

Pronóstico del gasto de hardware en tecnología de la información en todo el mundo de 2016 a 2021 (en miles de millones de dólares estadounidenses).

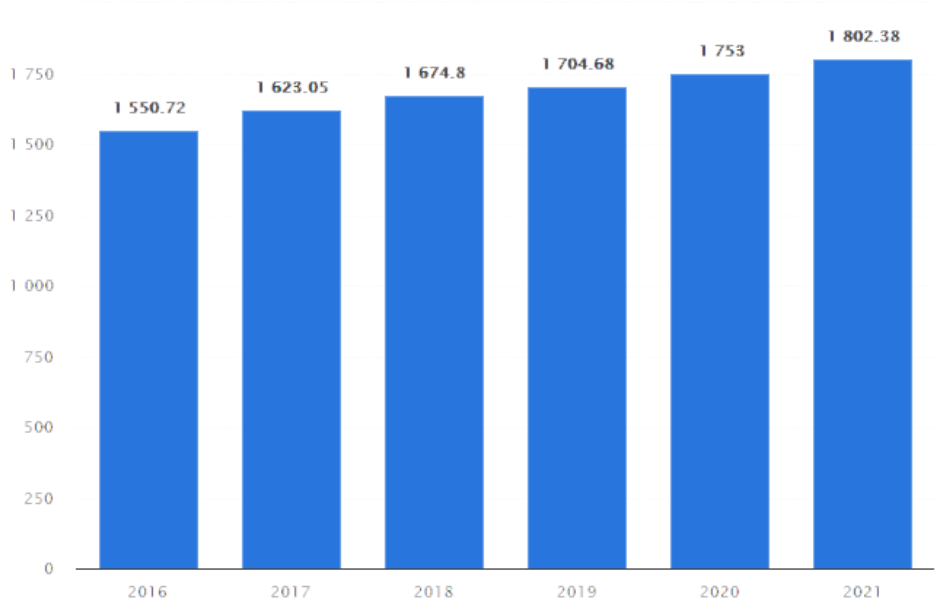

Figura 20 Ingresos del mercado de TI en todo el mundo desde 2016 hasta 2021 (en miles de millones de USD)

Anexo 6 Expertos encuestados

A continuación, se presenta la relación de los expertos a los cuales les fue aplicada la encuesta para la extracción de los valores de los atributos de calidad de la arquitectura.

Especialista	Rol	Años de experiencia	Tiempo en el puesto (años)	Categoría científica Certificaciones	Especialidad
1	Administrador de red	15	5	Ingeniero, Cisco CCNA	Cibernética
2	Administrador de red	16	8	Máster en ciencias, Cisco CCNA	Teleinformática
3	Administrador de red	16	16	Ingeniero, Cisco CCNA	Teleinformática
4	Administrador de red	16	5	Ingeniero, Cisco CCNA	Teleinformática
5	Subdirectora de centro de desarrollo	12	4	Ingeniera, Cisco CCNA	Ciencias Informáticas
6	Desarrollador	2	2	Ingeniero	Ciencias Informáticas
7	Desarrollador	4	4	Ingeniero	Ciencias Informáticas

Anexo 7 Bases de Datos de Gestión de Configuración

La gestión de la configuración se informatiza a través de la creación de una base de datos de gestión de configuración. El término base de datos de gestión de configuración CMDB (por las siglas en inglés de Configuration Management Database) proviene del marco de buenas prácticas ITIL, donde se define como una base de datos para almacenar los registros de configuración a lo largo del ciclo de vida de los servicios. La CMDB almacena información sobre: los elementos de configuración, las relaciones entre estos y los artefactos de procesos relacionados con los mismos (incidentes, problemas, solicitudes de cambios, acuerdos de nivel de servicio, etc). Los elementos de configuración, según ITIL, son todos aquellos que deben gestionarse para poder ofrecer un servicio TI a lo largo de todo su ciclo de vida e incluye: servicios, elementos de hardware, software, documentación de procesos, SLA, personas, etc. Las especificidades de qué almacenar y de los parámetros o atributos específicos que quedarán registrados de cada elemento, típicamente varían de una organización a otra.

Existen varias implementaciones de CMDB, dentro de los que se encuentra:

- ITOP
- Service Now
- Ansible CMDB
- Zendesk
- Ralph
- OTRS