Universidad de las Ciencias Informáticas

Facultad 2

Título:

"Propuesta de Arquitectura de un Emulador de Set Top Box para Plataformas IPTV"

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

Autor(es):

- ✓ Yandris Cajigal Méndez
- ✓ Angel López Henquen

Tutor(es):

- ✓ Ing. Darvis Dorvigny Dorvigny
- ✓ Ing. Serguei Guerra Fernández

Ciudad de la Habana, Junio de 2010

"Año 52 de la Revolución"

DECLARACIÓN DE AUTORÍA

Declaración de Autoría

Declaramos ser autores de la presente tesis y reconocemos a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmamos la presente a los __ días del mes de Junio de 2010.

Yandris Cajigal Méndez

Autor

Angel López Henquen

Autor

Ing. Darvis Dorvigny Dorvigny

Tutor

Ing. Serguei Guerra Fernández

Tutor

AGRADECIMIENTOS

Primero que todo agradecer a Fidel y la Revolución por permitirnos ser partícipes de este hermoso proyecto.

A nuestra familia por su amor, apoyo, dedicación, consejos, por ser nuestros guías.

A nuestros tutores por su apoyo incondicional, sus ideas, su entrega.

A todos nuestros amigos por los momentos compartidos.

A aquellas personas que de una forma u otra hicieron posible que este trabajo fuera un éxito.

Los autores.

Lo esencial es invisible a los ojos…

DEDICATORIA

A mi madre, por todo el amor, cariño, dedicación y apoyo que me ha brindado durante toda la vida.

A mi padre, por ser la gran guía de mi personalidad, quien junto a la educación inculcada por mi madre, formaron al hombre en que me he convertido hoy.

A mi hermana Yindra, por brindarme su cariño y su amor.

A mi hermana Yaselis, a la que le deseo muchos éxitos en la vida.

A mi abuela Eugenia, por ser la otra madre que me dio la vida y que nunca defraudaré.

A mi abuelo Juan, que con su ejemplo y enseñanza formó parte de la persona que soy hoy. ¡Este título es tuyo, abuelo!

A mi tío David, por su apoyo incondicional.

A todos mis tíos, tías, primos y amistades que siempre han querido lo mejor para mi.

A mí, por probarme que en la vida sólo hay que preponerse un objetivo y luego defender bajo cualquier circunstancia.

Yandris Cajigal Méndez

A mi familia por su armonía y unidad, a mi padre por ser indispensable y a mi madre por estar siempre presente.

Angel López Henquen

RESUMEN

Generalmente, no es necesario diseñar una nueva arquitectura de software para cada sistema. Lo habitual es adoptar una arquitectura conocida en función de sus ventajas e inconvenientes para cada uno de los casos en concreto. Hay que esforzarse para lograr esto y que se haga útil en la Universidad de las Ciencias Informáticas, definiendo una arquitectura propia para el desarrollo de los sistemas en cuestión.

En el presente trabajo de diploma se expone una propuesta de arquitectura de un emulador de Set Top Box para plataformas IPTV. Durante el desarrollo de la investigación se analizan las principales tendencias, tecnologías y metodologías actuales. Además se definen las funcionalidades principales del sistema y se plasman los requisitos no funcionales que complementan el funcionamiento del Set Top Box emulado. Se obtiene como resultado el diseño de la arquitectura para un sistema que permita emular un Set Top Box para plataforma IPTV, cumpliendo con las recomendaciones que brinda la Unión Internacional de Telecomunicaciones para un terminal de acceso IPTV.

Palabras Claves:
Arquitectura de Software, STB, IPTV.

ÍNDICE

INTRODUCCIÓN

Actualmente es notable en el mundo el enorme crecimiento en el uso de las Tecnologías de las Informáticas y las Comunicaciones (en lo adelante TIC). Inmerso en el vertiginoso desarrollo de la informática y las telecomunicaciones, el hombre trata de globalizar la información así como implementar herramientas cada vez más rápidas y eficientes, lo que convierte las computadoras no solo en una poderosa herramienta de trabajo, sino en una fuente de desarrollo del pensamiento humano; al contar entre otras facilidades, con la posibilidad de aumentar el caudal de conocimientos y la cultura general de cada cual mediante la difusión de contenidos multimedia. Todo esto viene acompañado del gran proceso de digitalización de los contenidos de imagen y sonido en el mundo de las comunicaciones, llegando en mucho de los casos hasta los hogares.

No se concibe en la actualidad prestar servicios de televisión (en lo adelante TV), voz y datos de forma aislada, por tanto la migración de las comunicaciones hacia la convergencia de los servicios conlleva al surgimiento del concepto de la tecnología "*Triple Play*", la cual pretende integrar los servicios antes mencionados sobre una misma red, en este caso la regida por los estándares de protocolos TCP/IP. El acrónimo IPTV (*Internet Protocol Television*) representa, por tanto, una tecnología que emerge, y puede cambiar la manera en que obtenemos entretenimiento en casa, operamos nuestros ordenadores personales así como usamos nuestros teléfonos móviles.

IPTV se refiere entonces al uso de las redes basadas en protocolo IP para distribuir servicios multimedia, tales como televisión, video, audio, texto, gráficos y datos, administrados estos para soportar los niveles de calidad del servicio, seguridad, interactividad y fiabilidad requeridos. En ocasiones el término IPTV es asociado a sistemas de distribución por suscripción de señales de televisión y de video bajo demanda; sin embargo es más que eso, ya que además de transformar los sistemas de televisión y radio actuales, son elementos influyentes en la informatización de la sociedad, en especial los sectores de la sociedad cubana que interactúan y conviven en nuestra universidad.

La tecnología IPTV requiere de una conexión a internet de banda ancha, mediante la cual se pueden conectar ordenadores personales, televisores IP, o televisores convencionales. Estos últimos para

acceder a los contenidos que se publican a través de la red de datos, necesitan de un dispositivo especial que se encarga de decodificar las señales que llegan en paquetes IP, y acomodarlas de manera que el televisor pueda interpretarlas y reproducirlas en forma de imagen y sonido. Este dispositivo, denominado comúnmente *Set Top Box* (en lo adelante STB) es indispensable para el disfrute de la tecnología IPTV, pues se encarga además de gestionar al usuario las diferentes funcionalidades y servicios que se ofertan.

Originalmente a los STB se les denominaba receptor de cable, y su misión era recibir la señal de las redes de cable (de codificación analógica), y servir la señal a un receptor de televisión. Posteriormente evolucionaron para realizar la decodificación de señales digitales recibidas por cable, antena terrestre o de satélite, por lo que habitualmente se asimilan a lo que se conoce como decodificador de televisión digital. Un STB es un equipo de acceso el cual principalmente se encarga de recibir una señal digital, en alguno de los medios tradicionales (cable, satélite, terrestre, y actualmente, la red de datos IP), comprobar que se tenga permiso para ver esta señal y transformarla para poder ser visualizada en un televisor.

La UCI ha dado sus primeros pasos en introducir la tecnología IPTV, procurando el desarrollo de una plataforma de este tipo al grupo IPTV de la facultad 2. Se hace visible la necesidad de contar con los dispositivos de recepción o STB para el despliegue y consumo de los servicios de dicha plataforma una vez terminada.

Al realizar un estudio acerca del estado de los STB a nivel mundial se detectaron una serie de problemas a tener en cuenta.

- ✓ Existen problemas de integración entre los STB y las Plataforma IPTV disponibles debido a que los fabricantes generalmente no se acogen a estándares globales al desarrollar los mismos, sino que desarrollan funcionalidades y características propias destinadas a cubrir sus necesidades.
- ✓ Los costos de los STB son muy elevados si tenemos en cuenta que esta tecnología es novedosa y por tanto acarrea altos valores de adquisición debido a su demanda.
- ✓ Es importante considerar la posibilidad de enviar los servicios IPTV a dispositivos que incluyan el hardware referido y sólo necesitan implementar funcionalidades. Esto nos lleva a buscar

alternativas en los STB emulados lo que nos genera nuevos problemas debido a que generalmente los STB son software propietarios enfocados a estándares específicos y con funcionalidades destinadas a cubrir requisitos ajenos a nuestras necesidades

Además es notable el impacto de la tecnología IPTV en la UCI si tenemos en cuenta las carencias en los servicios semejantes a los disponibles en una plataforma IPTV, servicios que brinda el sitio Inter-nos a nuestra comunidad universitaria tales como:

- ✓ No contar con un canal de retorno, lo que resta interactividad a la transmisión.
- ✓ No permite un servicio personalizado, pues no controla ni gestiona sus servicios ni usuarios.
- ✓ No implementa un medio que le permita garantizar la calidad de servicio.
- ✓ No incorpora servicios de valor agregado como pueden ser: Guía Electrónica de Programación, cambio de canal rápido, video-telefonía, ocio, comunicación, descarga de contenidos, etc.

Todo lo antes mencionado nos lleva a formular el siguiente **problema científico:** ¿Qué modelo de arquitectura utilizar sobre software libre que nos brinde la gama de servicios de una plataforma IPTV a través de un ordenador personal?
Para darle solución al problema anterior se propone como **objeto de estudio:** las arquitecturas existentes para el desarrollo de emuladores software de Set Top Box y como **campo de acción:** los procesos de gestión de un STB en la plataforma IPTV a desarrollar en la UCI.
Dado el problema a resolver se plantea como **objetivo general:** elaborar una propuesta de arquitectura que permita emular los procesos de gestión de un STB para la Plataforma IPTV a desarrollar en la UCI.

Para complementar el objetivo general antes trazado se proponen las siguientes tareas de la investigación:

- ✓ Análisis de la bibliografía referente al funcionamiento de los STB.
- ✓ Análisis de los sistemas informáticos existentes que emulan a los STB basados en hardware.

- ✓ Análisis de las tecnologías y arquitecturas definidas en las contribuciones y recomendaciones de la Unión Internacional de Telecomunicaciones (en lo adelante UIT) para plataformas IPTV.
- ✓ Valoración de las tendencias actuales y las tecnologías que se utilizan para construir software de este tipo.
- ✓ Selección y fundamentación de las herramientas a utilizar.

Este documento está dividido en 3 capítulos:

Capítulo 1. Fundamentación Teórica.

En éste capítulo se describen algunos de los conceptos relacionados con la plataforma IPTV, se hace énfasis en el funcionamiento de los Set Top Box para una plataforma IPTV, así como el análisis de soluciones existentes en el ámbito nacional e internacional. Además se describen características de las herramientas y metodologías más usadas en la realización de emuladores de Set Top Box.

Capítulo 2. Arquitectura de Software.

En este capítulo se analizan las tendencias, tecnologías, herramientas y línea base de la arquitectura a seguir para el diseño de la propuesta de arquitectura a plantear. Además se muestran los elementos fundamentales de la arquitectura base para los Set Top Box.

Capítulo 3. Propuesta de Arquitectura.

En este capítulo se plasma la propuesta de arquitectura que soporta el funcionamiento de un STB emulado para plataforma IPTV, cumpliendo con los requerimientos y contribuciones manifestadas en la arquitectura base definida. Además se evalúa la arquitectura partiendo de varios atributos de calidad.

CAPÍTULO 1: FUNDAMENTACIÓN TEÓRICA

1.1. Introducción

En éste capítulo se muestra una breve descripción de algunos de los conceptos relacionados con la plataforma IPTV, haciendo énfasis en el funcionamiento de los Set Top Box. Se hace un análisis de soluciones existentes en el ámbito nacional e internacional. Además se describen características de las herramientas y metodologías más usadas en la realización de emuladores de Set Top Box.

1.1.1. Televisión Digital.

La televisión es un sistema para la transmisión y recepción de imágenes en movimiento y sonido a distancia. Esta transmisión puede ser efectuada mediante ondas de radio o por redes especializadas de televisión por cable. El receptor de las señales es el televisor.

El consumo de contenido digital multimedia se está tornando cada día más común entre los usuarios de los servicios de telecomunicaciones. La digitalización de las imágenes posibilita su transmisión por medios electrónicos diseñados para el transporte de información.

En los sistemas de difusión de contenidos de audio y video digitales la información fluye desde servidores de contenidos hacia los terminales de los usuarios, de forma que tan pronto como la información es comprendida por el terminal comienza su reproducción. Esto significa que la información es reproducida a medida que va llegando, con un muy pequeño retardo respecto a la emisión del contenido por la fuente. Es muy similar a la distribución de audio y video analógicos, puesto que en estos casos la señal es reproducida por el terminal (TV estándar) tan pronto es recibida, no habiendo prácticamente retardo entre la recepción y la reproducción de la imagen.

1.1.2. ¿Cuál es la diferencia con la televisión actual?

La televisión actual es analógica y es aquella en que los niveles eléctricos varían en forma continua, sin interrupciones. La televisión digital se basa en el muestreo de voltajes, tomando una muestra a intervalos iguales. Cada muestra se transforma en un número digital binario que corresponde con el nivel de voltaje que tenía la señal analógica donde se tomó la muestra.

Esa muestra digital se transmite y se recibe en el receptor, donde ese número digital es reconvertido nuevamente en la señal analógica original, permitiendo someter la señal a procesos muy complejos, sin

degradación de calidad, ofreciendo múltiples ventajas y abriendo la posibilidad de nuevos servicios en el hogar.

La TV digital (en lo adelante TVD) se ha estado desarrollando intensamente en los últimos 10 años, resultando en la implementación de cuatro estándares de TV adoptados por distintos países del mundo. Las normas de TVD permiten el transporte de información de audio y video en tiempo real, reemplazando las normas de video analógico PAL, NTSC, SECAM, etc. En varios países ya se transmiten en forma regular señales de TVD terrestre o por cable, y se da prácticamente una cobertura global de TV digital vía satélite. Casi todos los países que hoy cuentan con emisoras de TVD terrestre están realizando planes para la migración total de los servicios de la TV conocida como terrestre de analógica a TVD en una cierta cantidad de años. La enorme eficiencia de la TVD posibilitará un mejor uso del espectro de radiofrecuencia, el que hoy se está constituyendo en un recurso escaso para una mayor eficiencia de las tecnologías de comunicaciones personales masivas.

El advenimiento de la TVD supone una nueva tecnología de transmisión totalmente incompatible con sus predecesoras. Esto significa que el usuario deberá reemplazar su TV estándar por uno con receptor de TVD o instalar un dispositivo convertidor de la señal para que esta sea convertida del nuevo formato a uno de los formatos pre-existentes.

Dado que la transmisión de la información utilizada para obtener una imagen de televisión y el sonido asociado, se realiza como “bits de datos”, las empresas de difusión digitales pueden entregar más información de la que es normalmente posible con tecnología analógica, por el mismo ancho de banda que actualmente ocupa la radiodifusión televisiva analógica (6 MHz u 8 MHz dependiendo de la zona).

Lo anterior en un contexto donde la televisión tradicional terrestre enfrenta un escenario complejo, frente a la oferta de multicanales por medio del éxito e importante penetración del cable y en menor medida de la televisión satelital, y el desarrollo de las empresas de telecomunicaciones que intentan posicionarse como nuevos entrantes en el negocio de la difusión de contenidos audiovisuales.

En los campos tecnológico y económico, el proceso de convergencia ha implicado en cierta forma la confluencia de las industrias audiovisual, de telecomunicaciones, de informática y de contenidos.

La TVD terrestre (en lo adelante TVD-T) afecta a todos los ámbitos del proceso televisivo, desde la producción hasta la transmisión y recepción de las señales televisivas. Lo anterior obliga a digitalizar todo el proceso, con una renovación de los equipos de producción y de transmisión (a cargo de los operadores) y de los equipos de recepción (a cargo de los usuarios).

En términos generales, la TVD permite un mejor uso del espectro de radiofrecuencia para la transmisión de las señales. Si la modalidad analógica implica el uso de una portadora modulada con 6 u 8 MHz de ancho de banda para la transmisión de una sola señal, con la digitalización y compresión de los canales, es posible transmitir varios canales de calidad igual o incluso superior a DVD en igual ancho de banda. Es importante resaltar la diferencia entre TVD e IPTV. La TVD se vale de la capacidad de un medio de difusión (terrestre, cable de TV o satelital) para propagar a múltiples usuarios básicamente la misma propuesta de contenido multimedia. Las redes de TVD están pensadas básicamente para la difusión de los contenidos. Estos contenidos pueden estar codificados de modo que solo los suscriptores que pagan por ellos tengan la posibilidad de visualizarlos al contar con la clave adecuada para su decodificación.

1.1.3. Evolución hacia la TVD y la IPTV.

La TVD consiste en la digitalización de la señal de TV analógica. Esta digitalización llegó primero a la TV satelital (desde hace más de una década), posibilitando la mejora de la oferta de la TV satelital previamente analógica.

La digitalización de la TV satelital se hizo mayoritariamente por medio del estándar DVB-S (*Digital Video Broadcasting* para Satélite), y al digitalizarse mejoró la calidad de la imagen (permitiendo reducir el tamaño de las antenas receptoras también), y posibilitó la difusión de múltiples canales con distintas modalidades de cifrado. En algunas zonas la TV satelital constituye prácticamente la única oferta de TV. Un soporte similar posibilitó la generalización de la TV por cable, puesto que la digitalización posibilitó receptores más económicos y eficientes viabilizando emisoras de cables aún en poblaciones pequeñas.

En lo que respecta a la digitalización de la TV cable, por tratarse de un servicio por suscripción, la evolución a digital es el resultado de una decisión del operador que debe ser acompañada de la adecuación de la infraestructura de recepción en el hogar del consumidor. La digitalización de la TV cable se está produciendo gradualmente, y generalmente apunta en primer lugar al segmento de clientes de mayor poder adquisitivo puesto que el atractivo de la propuesta es la calidad de imagen mejorada (similar a DVD, con múltiples canales de sonido) y la multiplicidad de nuevos canales que se transmiten cifrados (encriptados), posibilitando distribuir contenido más costoso para la modalidad de pago por visualización. Las operadoras de cable se han ido introduciendo a la TV interactiva por medio de un canal para retorno de datos, el que puede implementarse sobre una línea telefónica (cada vez más evitado), o sobre el

upstream IP que las operadoras de TV cable implementan para su servicio de provisión de acceso a Internet (cable-módem).

La TV de difusión terrestre es la que más debate técnico ha ocasionado, puesto que su modelo típico es de difusión abierta y gratuita, por lo que no existe una relación "cliente-proveedor" ni una contraprestación económica que pueda justificar que el proveedor reemplace equipamiento en el hogar de los consumidores. En este escenario son los gobiernos, responsables de conceder las licencias y espectro para las emisoras de difusión de TV, quienes tienen que coordinar el proceso de evolución de la TV analógica a digital, determinando al comienzo del mismo cuál será el modelo tecnológico a seguir.

Es posible distinguir cuatro aproximaciones diferentes a la TVD: el modelo estadounidense (ATSC); el modelo europeo (DVB), el modelo chino (DTMB) y el modelo japonés (ISDB) que se plantea en cierta forma como una combinación de los anteriores.

- El modelo estadounidense ATSC *(Advanced Television Systems Committee*), es impulsado por un comité formado por 140 empresas del área de radiodifusión y distribuidores de equipamientos electrónicos. La industria audiovisual y la industria manufacturera de equipos norteamericana, vieron en la aproximación de TVD terrestre de alta definición, una forma de establecer una diferenciación con respecto a otras ofertas multimedia en la era de la convergencia, y mantener una audiencia significativa a partir de un servicio televisivo tradicional de mayor calidad técnica.
- Por su parte, el modelo europeo DVB (*Digital Video Broadcasting*), impulsado por un consorcio de aproximadamente 270 empresas de radiodifusión y distribuidores de equipamiento europeos (tales como Nokia, Siemens y BBC, entre otros), plantea la promoción del uso de la capacidad adicional para proveer más contenidos televisivos y nuevos servicios de información. La transmisión de múltiples señales de información multiplexadas en un mismo canal, se sustenta en la posibilidad de proveer TV multicanal a una fracción importante de usuarios. El proyecto de TVD europeo apunta al desarrollo de un aparato de recepción multimedia de servicios integrados. El estándar prevé la existencia de un canal digital de retorno sobre el cual se sustenta la interactividad.
- El modelo japonés ISDB (*Integrated Service Digital Broadcasting*), es defendido por las grandes redes de ese país, y fue adoptado como la norma de TV digital de Brasil. Es una combinación entre los dos modelos anteriores. Atendería a los requisitos de la alta definición, pero también ofrecería la posibilidad de transmitir con una definición estándar, con calidad inferior, para permitir

una programación múltiple. También posibilita, de forma similar a DVB, la transmisión de TV a terminales móviles.

- DTMB *(Digital Terrestrial/Television Multimedia Broadcasting*). Desarrollado en la República Popular China, aprobado en agosto de 2007, con características diferentes a los otros estándares tanto en el sistema de modulación como de codificación de canal y en el que se funden dos estándares previos también desarrollados en China, ADTB-T, similar al ATSC y el DMB-T.

La adopción de la norma de TVD para difusión terrestre se ha tornado en una cuestión de decisión nacional en la mayoría de los países del mundo, y los gobiernos han intentado compatibilizar la viabilidad de la industria de difusión de contenidos con el interés de los consumidores, definiendo su proceso de transición de TV analógica terrestre a TVD.

1.1.4. ¿Cuáles son los beneficios de la Televisión Digital?

- ✓ **Calidad**.

Aumenta la nitidez, resolución de la imagen y la calidad del audio, debido a que la transmisión digital no se ve afectada por interferencias y ruidos.

- ✓ **Movilidad y Portabilidad.**

Permitirá la recepción del servicio en dispositivos móviles (celulares, televisores portátiles entre otros), en óptimas condiciones.

- ✓ **Optimización del uso del espectro.**

En el mismo ancho de banda de 6 MHz que con tecnología analógica se transmite video y audio, en digital se puede transmitir varias programaciones diferentes en alta definición y definición estándar, así como datos adicionales de información particular o general.

- ✓ **Interactividad.**

Permite integrar los contenidos de televisión, tanto a través de servicios públicos como servicios comerciales, que hasta ahora sólo eran accesibles a través de otros servicios.

1.2. ¿IPTV o TV en Internet?

A menudo es utilizado el término IPTV para hacer referencia a la transmisión de TV en Internet. Sin embargo es notable la diferencia entre ambos conceptos.

La TV en Internet es un servicio abierto, dirigido al público en general, con principio de distribución gratuita, cuya función social está orientada al entretenimiento, cultura e información. Existe un gran número de productores medianos y pequeños que contribuyen con creaciones innovadoras en contenidos y canales, distribuidos típicamente sobre redes IP embebiendo el flujo de video en los sitios web.

En cambio IPTV se aplica a redes de telecomunicaciones de banda ancha mediante sistemas propietarios y de pago, cuyo objetivo fundamental es el control de flujos y usuarios, explotando al máximo el ancho de banda disponible y brindando servicios mucho más personalizados. En contraste con la TV en Internet, los contenidos se distribuyen mediante una infraestructura de red cerrada, basada en el protocolo IP, pero codificando los contenidos según los niveles de control y calidad gestionados por el proveedor, utilizando protocolos de comunicación más especializados en el envío y recepción de contenidos multimedia tanto en difusión de TV como en servicios de VoD.

Aunque ambos se basan en la entrega de video sobre el protocolo IP, sus enfoques en esta tecnología difieren en varios aspectos. [1]

Diferentes Plataformas.

Como su nombre lo sugiere, TV en Internet aprovecha la Internet pública para hacer llegar el contenido a los usuarios finales.

IPTV en cambio utiliza redes privadas, dedicadas y seguras para el envío de contenido a sus clientes. Estas redes privadas son administradas por los proveedores de servicios IPTV. [1]

Alcance geográfico.

Las redes privadas que implementan los operadores de IPTV no son accesibles a los usuarios de internet y generalmente se encuentran en zonas geográficas fijas.

En este caso TV en Internet no tiene limitaciones geográficas, por tanto a sus servicios de televisión se puede acceder desde cualquier parte del mundo. [1]

Propiedades de la infraestructura de red.

Cuando el video se envía a través de la Internet pública, algunos de los paquetes IP portadores del contenido puede retrasarse o incluso perderse a medida que atraviesan las sub-redes que conforman la red pública de Internet. Como resultado, los proveedores de contenidos de vídeo a través de Internet no pueden garantizar una experiencia de televisión que se compare al menos con la TV tradicional terrestre, mucho menos con la TV por cable o con la TV satelital.
De hecho, en el envío de vídeo a través de Internet el esfuerzo fundamental reside en entregar el contenido al usuario, por lo que habitualmente la resolución de la imagen es bastante baja.
En comparación con esta experiencia, la IPTV se entrega en una infraestructura de red que generalmente es propiedad del proveedor de servicios. Ser propietario de la infraestructura de red permite a los operadores de IPTV diseñar sus sistemas enfocados en entregar vídeo de alta calidad. [1]

Mecanismo de acceso.

Generalmente para el acceso al contenido de vídeo emitido a través de un sistema de IPTV se utiliza un STB, encargado de decodificar y ajustar el video al medio donde reproduzca el mismo.
Mientras que para acceder a los contenidos de TV en Internet se utiliza una PC, donde el tipo de software utilizado dependerá del tipo de contenido a visualizar. [1]

Resumiendo, IPTV está dominado por el control y calidad del producto, así como el alto grado de seguridad, mientras que TV en Internet se caracteriza por su variedad y cantidad de producciones.

Definitivamente algún día convergerán en un solo concepto.

1.3. Plataforma IPTV.

IPTV es una tecnología que emplea una red IP que garantiza la calidad de servicio para cada flujo de información de video, luego que podemos definir como IPTV "contenido de vídeo digital, incluyendo la televisión, que se realiza mediante el uso del Protocolo Internet. En el caso de la TV sobre Internet el flujo

de información hace uso de un servicio basado en el “mejor esfuerzo”. IPTV constituye una tecnología de TV interactiva que puede igualar o mejorar la calidad de la distribución de señales digitales de video que hoy implementan la TVD tanto sea en sus versiones terrestre y satelital. Algunos de los contenidos que IPTV reproduce pueden variar entre vídeos musicales, programas de televisión, películas, conciertos y una variedad de eventos especiales, tales como beisbol, partidos de fútbol etc. Esto significa que nuestra definición breve de IPTV abarca una amplia gama de actividades, potenciando la variedad de las mismas. Unos de esas actividades podrían incluir la descarga de una película, vídeo o música a través de Internet lo cual podría verse en el momento de la descarga o posteriormente.

1.3.1. Funcionalidades de la plataforma IPTV.

Una solución de IPTV se compone entre otras de las siguientes funcionalidades:

- Guía Electrónica de Programación (en lo adelante EPG) que tiene como función facilitar la visualización de la programación, la selección y la búsqueda de programas en los canales en vivo.
- Sistema de menús para la selección, búsqueda y compra de los videos bajo demanda.
- Sistema que permite la grabación por parte de los usuarios la programación en vivo (en lo adelante PVR) de modo que se pueda reproducir la misma posteriormente.
- Sistema que permite la repetición inmediata y congelado de escenas de un programa en vivo (*Time Shift TV - TSTV*).
- Cuenta con la capacidad de presentar las señales (de los canales en vivo) en Mosaico.
- Cuenta con un sistema para control parental y para compra de servicios como el de Video/Audio a demanda.
- Ofrecen el contenido con audio de alta calidad (por ejemplo *Dolby Stereo* y *Dolby 5.1*).
- Soportan servicios interactivos desarrollados por el operador o terceras partes que interactúan con el operador por medio de interfaces abiertas.
- Posibilidad de contar con otros servicios interactivos como: multilenguaje (subtítulos y audio), compras, integración con otras plataformas de mensajería, capacidad de navegación por internet en el TV, videoconferencia, etc.

1.3.2. IPTV como servicios.

Las funcionalidades mencionadas anteriormente vienen asociadas a servicios específicos de la plataforma IPTV, que por su relevancia en la misma a continuación se puntualizan.

1.3.3. Servicio de Difusión de TV.

Este es el caso en el que el usuario posee el acceso a un número determinado de señales de video de manera similar al servicio ofrecido de TV por cable en la actualidad.
Para este tipo de servicio la velocidad del cambio de canales es más lenta que en el servicio de TV por cable tradicional, debido a dos factores que se interponen:

- La necesidad de que el STB seleccione un grupo de multidifusión correspondiente al canal que se desea recibir la señal y libere al grupo anterior que tenía seleccionado. Esta operación puede demorar varios milisegundos.

- La detección de las tramas de video que vienen comprimidas que permiten reconstruir la imagen, de tal manera que puede esto ocurrir cada varios segundos. Este aspecto es el principal contribuyente a la lentitud del cambio de canales.

En dependencia del tipo de equipo de acceso (STB) que se instale en la casa del cliente, al ancho de banda disponible en la interfaz de la Línea de Suscripción Digital Asimétrica (en lo adelante ADSL) y al tipo de acceso de cobre el cliente podrá acceder a una, dos o tres señales de video simultáneamente.
El cliente también podrá configurar su servicio, determinando que paquetes de señales quiere recibir, si desea recibir algún tipo de señal que no está incluido en el paquete contratado previamente.

1.3.4. Servicio de video bajo demanda (VoD).

El cliente dispone de una lista de películas, eventos deportivos en vivo, entre otros, de los cuales podrá elegir el que desee para ver en un momento determinado.

Para este tipo de servicio existen dos variantes:

- **VoD en tiempo real:** El cliente solicita la señal de video, la recibe inmediatamente lo que genera que en la red se ocupe, entre el servidor de video y el cliente, el ancho de banda necesario para la transmisión de la misma. Este tipo de servicio carga la red de transporte en forma directamente proporcional a la cantidad de clientes que estén utilizando este tipo de servicio.

- **VoD en espera:** Es la variante del servicio VoD en tiempo real, pero se limita al comienzo del contenido de video a horas determinadas (esta puede variar según un tiempo predeterminado por

el proveedor de servicios IPTV). Esto permite limitar el ancho de banda requerido en la red, ya que todos los clientes que comenzaron a ver contenido de video a la misma hora ocupan en la red solamente el ancho de banda del contenido de video en cuestión.

1.3.5. Acceso ADSL.

Estos servicios de transmisión de video y multimedia, en general son soportados sobre acceso de banda ancha ADSL o ADSL2+, sobre el par de cobre que llega al cliente. Se pueden alcanzar velocidades teóricamente de hasta 8 Mbps en el caso de ADSL y 24 Mbps en el caso de ADSL2+, lo que es realidad se ve disminuido por afectaciones tales como las interferencias de otros servicios de datos o atenuaciones debidas a la longitud de los cables o el estado técnico de los mismos.

Para brindar este acceso se necesita contar con un módem ADSL o ADSL2+ en el domicilio del cliente, el cual se conecta mediante el par de cobre a un equipo de acceso de datos denominado DSLAM, ubicado en las centrales telefónicas. Desde el DSLAM se llega mediante el núcleo de datos a los dispositivos centrales de la solución, como son servidores de video, *middleware* y codificadores. Los módems ADSL mencionados tienen más de un puerto Ethernet de manera que puedan servir al STB, a la PC o a la red interna del domicilio del abonado, de lo contrario se necesitará un dispositivo adicional como un *Hub* o *Switch* a la salida del módem al que se conectarán el STB y la PC.

En caso de existir un Firewall, se debe proporcionar algún mecanismo para resolver problemas tales como *Network Address Translation (*en lo adelante NAT).

Las principales características a tener en cuenta para este tipo de elemento son las siguientes:

- Longitud y estado de los cables de la planta externa (red de acceso metálica)
- El acceso ADSL debe proveer el ancho de banda suficiente hasta la casa del cliente y debe poder asegurar una determinada calidad de servicio.
- Una consideración adicional a la hora de determinar la escalabilidad del DSLAM, es la cantidad de ancho de banda que puede recibir.

- En caso de DSLAM Ethernet, estos deben incluir:
  - Funcionalidad de IGMP *Snooping* o IGMP Proxy.

- Buena velocidad de conmutación de canales de multidifusión hacia los usuarios, cuando hay varios usuarios realizando cambio de canales y se utilizan mecanismos de "IGMP *Proxing*" o "IGMP *Snooping*".
- Poder equiparse con una o más interfaces de red (ascendente) Gigabyte Ethernet de modo de tener suficiente ancho de banda según las necesidades.
- Manejo y mapeo de VLAN (IEEE 802.1Q) para diferenciar tráfico de diferentes servicios (video, internet, etc.)
- Priorización de tráfico según bits de prioridad 802.1p, para priorizar servicios de tiempo real como video frente a servicios de datos.
- Limitación de la tasa de datos en las interfaces de subida por VLAN, etc.
- Limitación de la tasa de datos en las interfaces hacia los usuarios por PVC, por VLAN, etc.

➢ Los Módems deberán corresponder por ejemplo a alguna de las siguientes posibles alternativas:

- Que el módem disponga de varios puertos Ethernet para facilitar la integración de varios STB o PCs en la casa del cliente. El módem debe diferenciar los servicios de cada puerto por PVC o VLAN diferentes hacia la red. El módem deberá priorizar el tráfico hacia la red en función del PVC o de la VLAN.
- Que esté el módem incluido dentro del STB y con varios puertos para varios STB y PCs en la casa del cliente. El módem/STB debe diferenciar los servicios de cada puerto por PVC o VLAN diferentes hacia la red. El módem/STB deberá priorizar el tráfico hacia la red en función del PVC o de la VLAN.
- Que esté el módem incluido dentro del STB y con varios puertos para varios STB y PCs en la casa del cliente. El módem/STB debe diferenciar los servicios de cada puerto por PVC o LAN diferentes hacia la red. El módem/STB deberá priorizar el tráfico hacia la red en función del PVC o de la VLAN.
- Utilizar un módem con un único puerto y un *switch* en la casa del cliente. El *switch* debe diferenciar los servicios de cada puerto por VLAN diferentes hacia la red. El *switch* o el módem deberán priorizar el tráfico hacia la red en función de la VLAN que corresponda.

➢ Es conveniente que el módem pueda limitar la tasa de tráfico hacia la red por PVC y por VLAN para aplicarse según corresponda.

1.3.6. Middleware.

El middleware es una plataforma de gestión de aplicaciones que interactúa con la red de acceso y los STB para permitir el aprovisionamiento y la distribución de servicios de televisión interactivos. Es una parte crítica de la solución IPTV extremo a extremo, constituye una especie de "cerebro" de la red. Es una plataforma informática que provee las herramientas y tecnologías necesarias a los proveedores de servicios de banda ancha para ofrecer servicios de video haciendo uso de su infraestructura. El middleware interactúa con los STB por lo que ambos deberán contar con aplicaciones diseñadas en la modalidad cliente (en el STB) y servidor (en el middleware).

El middleware es el responsable de asegurar la completa interoperabilidad del servicio de video. No está limitado a una única operación en el sistema, pero debe ser capaz de comunicarse directamente con cada componente para proveer soluciones de video. Esta entidad es la que realiza el control y la operación de la red, en el sentido de que es la que gestiona los servicios y los usuarios según se muestra a continuación.

En el servicio de VoD, el usuario que desea ordenar una película para ser vista en determinado día y hora, realiza la solicitud al middleware, estableciendo una sesión con éste. El middleware lo autorizará o no en función de la cuenta del usuario, su perfil, etc. En caso afirmativo le ordenará al servidor de video que realice la transmisión de la película en el horario solicitado por el cliente.

Usualmente el middleware y el STB funcionan en modalidad servidor – cliente. Es por ello que puede existir incompatibilidad entre ciertas soluciones de middleware y STB de fabricantes distintos.

El componente del middleware en el *Back-Office* (Servidor) corre en los servidores de aplicación, típicamente ubicados en el *Head-End*. El componente cliente del middleware corre en el STB del suscriptor. Por lo general se debe cargar la aplicación cliente en el sistema operativo de STB a medida del middleware a utilizar.

Las principales características que poseen los middleware son:

- El middleware provee las *Application Programming Interface* (en lo adelante API) necesarias para extender la habilidad de desarrollar nuevas funciones y permitir el pasaje de datos entre sistemas de terceras partes o ya existentes en la red del operador.
- Los servicios básicos que el middleware debe soportar incluyen difusión de TV, EPG, *Pay-per-View*, y VoD.
- Otras características más avanzadas incluyen suscripción de VoD y PVR, gestión de clientes, autoconfiguración de los STB, auditoría y trazabilidad, auto aprovisionamiento de clientes, gestión automatizada de contenido, y creación automatizada de EPG.
- La escalabilidad o la habilidad de soportar un gran número de clientes, es también una característica esencial del middleware.
- La interoperabilidad con diferentes STB es un punto a tener en cuenta, en general los fabricantes listan los STB soportados.
- Esta es la entidad funcional de la red de video menos estándar, es decir varían bastante de fabricante en fabricante las funcionalidades que posee, por lo que es un tema importante analizar qué funciones incluye cada uno. Por ejemplo: en algunos casos incluye un servidor DHCP, en otros se comunica con uno. Algunos equipos incluyen también en el middleware la entidad de autenticación, en otras se comunica con un servidor existente. Otra variable es como se maneja la base de datos de los usuarios, con sus perfiles y demás, que tipo bases de datos, etc.

Se entenderá por middleware al conjunto de aplicaciones de software ejecutándose en una cierta arquitectura de servidores, con el cometido de soportar la entrega de servicios de IPTV. El middleware define y coordina la forma en que el usuario interactúa con el servicio de IPTV. [2]

El middleware hace disponible a los usuarios una interfaz gráfica amigable y configurable, para acceder y controlar los servicios y funcionalidades mencionados. También deberá disponer de un módulo de tasación que permita la asignación de precios a los consumos a demanda, y gestiona las suscripciones al servicio.

1.3.7. Interfaces.

El middleware podrá contar con interfaces hacia el resto de los componentes de la solución (STB, sistema de DRM, servidores de video, sistemas de facturación y aprovisionamiento, etc.).

De manera particular se pueden encontrar:

- Interfaz con el sistema DRM/CAS.
- Interfaz con el sistema de facturación, el cual puede ser en tiempo real o no (ésta varía en dependencia del sistema de pago de los servicios).

1.3.8. Interfaz gráfica de usuario.

La Interfaz gráfica de usuario es presentada de forma tal que el usuario pueda utilizarla intuitivamente en todo su potencial. Brindan acceso a todo el sistema de VoD y a consultas sobre la programación disponible en difusión de TV mediante la EPG.

1.3.9. Protocolos de comunicación en la plataforma.

1.3.9.1. Difusión de TV mediante IGMP.

El protocolo de red IGMP se utiliza para intercambiar información acerca del estado de pertenencia entre enrutadores IP que admiten la multidifusión y miembros de grupos de multidifusión.

Cuando una aplicación en un host se suscribe a un grupo particular, el host envía un mensaje de informe con la dirección del grupo al que se ha suscrito. Periódicamente, los routers envían interrogaciones al grupo y cada ordenador responde con un informe por cada grupo al que pertenece, si un host observa un informe de algún otro host asociado al mismo grupo de multidifusión, no envía su propio mensaje. El servidor envía una única trama IP a todos los destinos que la están demandando en ese momento.

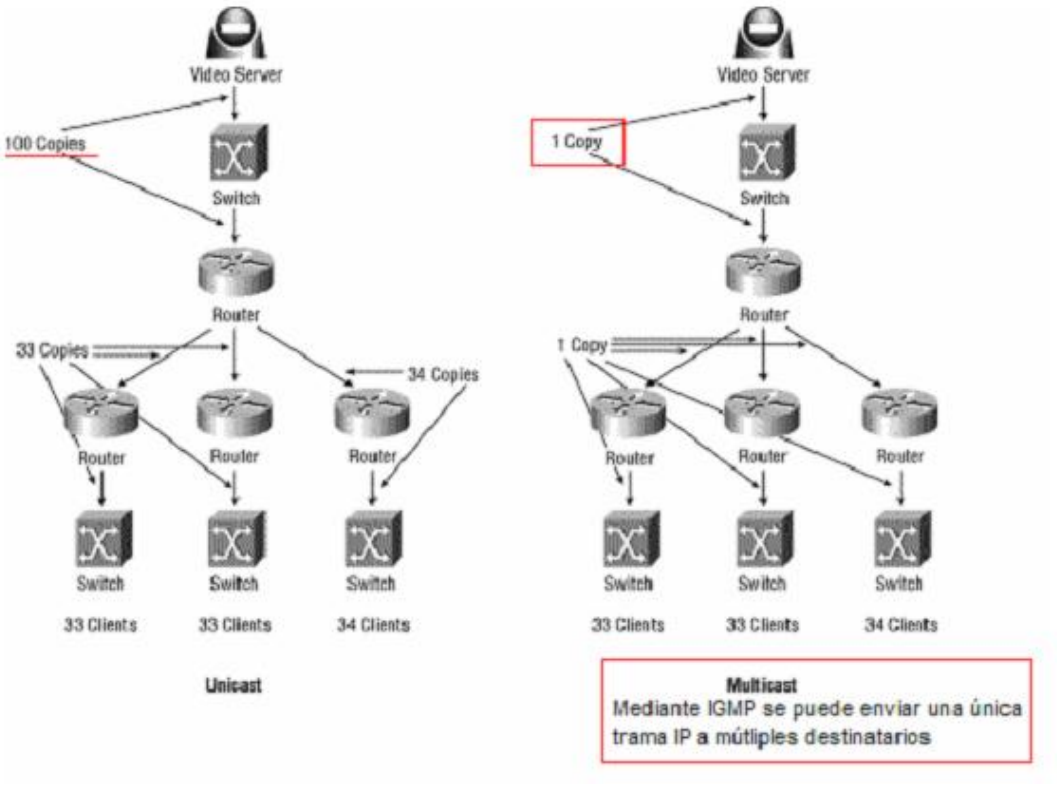


Figura.1 Funcionamiento IGMP.

1.3.9.2. Real Time Streaming Protocol (RTSP) y Session Description Protocol (SDP).

El protocolo de flujo de datos en tiempo real (RTSP) establece y controla uno o muchos flujos sincronizados de datos, ya sean de audio o de video. RTSP se usa para el establecimiento y control de la sesión de flujo de video, actuando como un mando a distancia de la sesión.

Se emplea en conjunto con SDP (Protocolo de Descripción de Sesión), que es el encargado de proporcionar información sobre la sesión: número de flujos, tipo de contenido, duración, ancho de banda, etc.

Características principales de RTSP.

- ✓ Protocolo de nivel de aplicación.
- ✓ Independiente de la capa de transporte (TCP o UDP).

- ✓ No es el encargado de transportar los contenidos.
- ✓ Un servidor RTSP necesita mantener el estado de la conexión.
- ✓ Capacidad multiservidor: Cada flujo multimedia dentro de una presentación puede residir en servidores diferentes.

1.3.9.3. Real Time Transport Protocol (RTCP).

RTCP es un protocolo de comunicación que proporciona información de control que está asociado con un flujo de datos para una aplicación multimedia (flujo RTP). Se usa habitualmente para transmitir paquetes de control a los participantes de una sesión multimedia de streaming.

Características principales de RTP.

RTP no ofrece garantías sobre la calidad del servicio ni sobre el retraso de la entrega de datos, estos deben ser proporcionados por la red subyacente. RTP ofrece entrega de datos multicast.

Por tanto, RTCP y RTP poseen cometidos diferentes. Mientras que el primero es el encargado del establecimiento y control de la conexión video-streaming, RTP se emplea para transportar los contenidos en tiempo real (audio y video).

Características principales de RTCP.

- ✓ Trabaja junto con RTP en el transporte y empaquetado de datos multimedia, pero no transporta ningún dato por sí mismo.
- ✓ Se emplea para monitorizar la calidad de servicio.

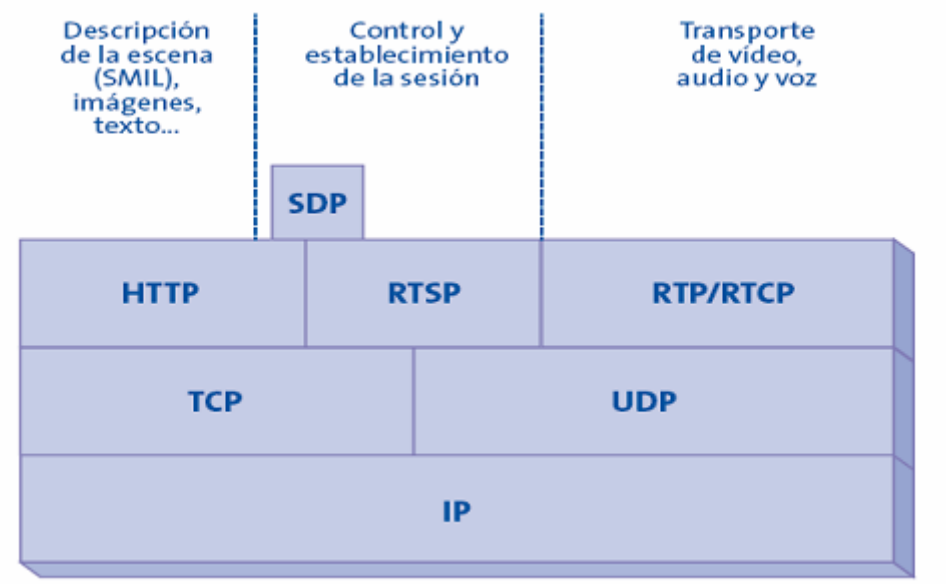


Figura.2 Escenario de Video – Streaming.

1.3.10. Seguridad en la plataforma IPTV.

1.3.10.1. CAS

Los sistemas de acceso condicional (CAS) contribuyen a garantizar que sólo los abonados autorizados tengan acceso a los contenidos, y así crear una protección contra el robo del servicio. La función del CAS se basa en el cifrado de la información. Una vez cifrada esta, el sistema garantizará que las claves de sesión sólo se envíen a los suscriptores autorizados a recibir el contenido llevando a cabo contantes verificaciones en la plataforma ante cada solicitud o petición del usuario [3].

1.3.10.2. DRM

Hoy es prácticamente imprescindible que una solución de IPTV cuente con un sistema de gestión de derechos digitales (DRM), ya que es el encargado de controlar la seguridad de los contenidos digitales que se distribuyen en la plataforma. Este sistema podría ser aplicado a los servicios VoD, AoD, difusión de TV y otros contenidos bajo demanda (ya sean almacenados o en vivo), de modo que se prevea la copia de la información digital y su posterior distribución.

El DRM implica un cierto cifrado del contenido multimedia, el cual luego puede ser reproducido si el receptor cuenta con la licencia correspondiente. La licencia es básicamente la clave para desencriptar el

contenido. Esto evita la copia del contenido digital que se distribuye, puesto que el mismo nunca deja de estar cifrado por medio del DRM. Si un suscriptor copia el contenido digital que recibe a otra persona, si ésta no cuenta con la clave para desencriptar el contenido no podrá reproducirlo en su sistema [3].

1.4. Set Top Box.

1.4.1. ¿Qué es Set Top Box?

Set Top Box, cuya traducción literal al español la interpretamos como dispositivo o caja encima del televisor, es el nombre con el que se conoce el dispositivo encargado de la recepción y decodificación de señal de TVD, para luego ser mostrada en un dispositivo de televisión. En el contexto de IPTV, es el STB el dispositivo intermediario entre el usuario y las diferentes funcionalidades y servicios que brinda la plataforma IPTV.

Originalmente a este tipo de aparatos se les denominaba receptor de cable y su misión era recibir la señal de las redes de cable (de codificación analógica) y servir la señal a un receptor de televisión. Posteriormente evolucionaron para realizar la decodificación de señales digitales recibidas por cable o antena terrestre o de satélite, por lo que habitualmente se asimilan a lo que se conoce como decodificador de televisión digital.

Un STB principalmente se encarga de recibir una señal digital, en alguno de los estándares (cable, satélite, terrestre, IPTV), y de comprobar que se tenga permiso para ver este señal. Posteriormente lo desmodula y lo envía al televisor.

Debido a que la mayoría de televisores en el mundo son analógicos se sobreentiende la importancia de este dispositivo, el cual será básico hasta se disponga de televisores digitales a un precio accesible en el mercado. Mientras tanto los consumidores que deseen acceder a los servicios de la televisión digital, necesitarán un STB para su recepción.

Los STB presentan una interfaz Ethernet hacia la red de datos, que en general estará conectada al módem ADSL. En algunos casos los propios STB cuentan con el módem ADSL integrado. A su vez esos STB presentan una interfaz RCA o coaxial hacia el televisor, algunos además cuentan con interfaces de datos hacia el usuario. Por lo general cuentan con un control remoto mediante el cual se seleccionan las distintas opciones, programación, etc., y en algunos casos incluyen o se les puede agregar un teclado inalámbrico el cual es útil en caso de realizar navegación en Internet desde la

TV. Cuenta con un pequeño procesador el que corre una aplicación llamada EPG generalmente sobre plataformas Linux o Windows CE. Ésta es la que da soporte para aplicaciones como un browser de configuración del servicio, suscripción a canales, orden de películas en servicio de VoD, visualización del catálogo de películas, control de acceso, etc.

Actualmente un STB puede ofrecer muchos servicios, desde utilizarlo como grabador en los STB que incorporen disco duro, como utilizarlos para hacer consultas meteorológicas, hacer la reserva de una visita médica, o hacer compras en los que disponen de interactividad. También muchos de ellos nos dan la opción de conectarles dispositivos externos como podrían ser videocámaras, impresoras, etc.

1.4.2. Principales características de los STB.

Las principales características a tener en cuenta para este tipo de elemento son las siguientes:

- Se debe considerar si se desea que el STB tenga disco duro incluido o no, esto es utilizado en algunas soluciones para aplicaciones tales como *PVR*, también es útil para funcionalidades tales como detener o rebobinar una película en VoD. Se debe considerar el alto costo de los STB con disco incluido, además del hecho de que los proveedores de contenido prefieren o imponen que el almacenamiento no se haga en el equipo del cliente por seguridad.
- Otro punto importante es el del acceso, si el STB cuenta con módem ADSL incluido, si tiene varias interfaces Ethernet, en este caso si funciona como *Hub* o como *switch*, si maneja herramientas para calidad de servicios (en lo adelante QoS) como 802.1p/q. Diferenciación, priorización y limitación del tráfico de los diferentes servicios que brinde el STB a través de sus puertos.
- En algunos casos los STB cuentan con una interfaz de datos hacia el usuario con Wi-Fi por ejemplo 802.11g de modo de que este implemente una red interna inalámbrica.

Es de gran importancia analizar los formatos de salida que se desean de acuerdo a los planes comerciales. Por ejemplo, si además de salidas de señal de vídeo analógica, ya sea *s-video* (video separado) o video compuesto RCA se desea salida RF, ésta no esta presente en todos los STB y puede ser necesaria ya que se cuenta con un gran parque de televisores solo con entrada RF. También es importante si se desea salida de audio digital o no, con formato 5.1 como para “Teatro en Casa”, etc. Estos requerimientos dependen fuertemente del segmento de mercado al que se apunte. Estas variantes tienen muchas posibles combinaciones e implican diferencias importantes en los precios.

1.4.3. Funcionamiento de los STB.

Los pasos que lleva a cabo un STB para complementar su funcionamiento son los siguientes:

- Lo primero que hace es sintonizar una señal digital, la cual incluirá tanto información de video (MPEG2, o MPEG4 para señales en alta definición), información de audio e información de datos.
- El siguiente paso es separar los tres tipos de información que recibimos para tratarlos por separado.
- A continuación, es sistema de acceso condicional decidirá cuáles son los permisos que tiene el subscriptor para poder ver los contenidos que esta recibiendo. Si tiene permiso descifrará esa información
- Una vez descifrados, los paquetes de video y audio son enviados al televisor.
- Los paquetes de datos que hemos recibido junto con los de video y audio, se ejecutarán en caso de ser necesarios o solicitados por el consumidor.
- El STB puede poseer un canal de retorno por donde enviar datos a la cabecera (*Back Channel*).

Para poder ejecutar los datos o programas descargados de la señal de datos, se necesitan una serie de elementos. Estos se pueden describir por el siguiente esquema de capas muy parecido al de un ordenador.

- *Capa de Hardware:* Son todos los componentes físicos que forman un STB (CPU, Memoria, acceso condicional, decodificador MPEG...)
- *Sistema operativo:* Al igual que en un ordenador, un STB también necesita de un sistema operativo para su funcionamiento. La diferencia básica sería en que un STB, necesita de un sistema operativo en tiempo real, ya que, operaciones como la decodificación MPEG necesitan que se realicen al instante. Algunos ejemplos de sistema operativo serian: Linux, Windows CE.
- *La plataforma o Middleware:* Se trata de una capa intermedia entre la capa *hardware* y la *software*. Se trata de un conjunto de módulos que permiten un desarrollo más eficiente de las aplicaciones. El middleware proporciona un API para cada uno de los tipos de programación que soporta. De los

diferentes lenguajes de programación que puede soportar un STB, el que sería más destacable, sería DVB-J, que es utilizado para las aplicaciones interactivas.

- *Capa de aplicaciones:* Aquí es donde encontraremos las aplicaciones, que una vez descargadas se podrán ejecutar (algunas aplicaciones podrían ser: EPG, anuncios interactivos...). A diferencia de las demás capas, ésta no debe de estar operativa en todo momento, pues simplemente se ejecutará cuando el consumidor lo solicite.

1.4.4. STB basados en Hardware.

IPTV en PlayStation 3

La PlayStation3 (PS3) es una máquina potente informática en sí, dotado de gran almacenamiento de datos, rápido procesador gráfico y un controlador de disco *Blue-Ray*.

La PS3 se puede usar como STB, inicialmente el primer servicio IPTV a través de PS3 se brindó en Corea con las empresas MEGA TV y KOREA TELECOMS. Es notorio el aumento de suscriptores para este tipo de dispositivos frente a los servicios de tráfico multimedia sobre internet en especial IPTV, dado el rendimiento que presentan estas máquinas frente a otros dispositivos del hogar.

Decodificador Digital CDR 1000D

Este Decodificador Digital compatible con la Norma Argentina de Televisión Digital le ofrece la posibilidad de ver Señales Digitales estándar, y posee también salida puerto HDMI para Alta Definición.
Entre sus características más destacables debemos enunciar que soporta televisión digital interactiva, además posee un navegador web que nos permite navegar en internet.

1.4.5. STB basados en Software.

Emulador STB BCI.

Se trata de una solución basada en software que simula los mensajes de control RTSP de un Set Top Box. Usando la interfaz el usuario puede seleccionar varias sesiones de RTSP, dependiendo de los proveedores de servicios y las posibilidades de red que posea. El usuario simplemente ingresa el número

de sesiones a las que quiere acceder luego el sistema generará el número apropiado de sesiones. Cada sesión puede tener incluido los servicios *Time Shift*.

Ventajas:

- ✓ Elimina la necesidad de utilizar varios STB.
- ✓ Se puede ejecutar de manera local o remota.
- ✓ Facilidad de uso desde la computadora local hacia la los servidores a gran escala.
- ✓ Fácil instalación y ahorro de costos de hardware.

Uso:

Esta herramienta tiene como por objeto proporcionar un medio simple de simulación de los STB tradicionales. Su simulación se maneja de manera efectiva a través de una interfaz gráfica de usuario. Las sesiones pueden ser establecidas y controladas por un ordenador portátil a través de la red. De esta forma el usuario puede ampliar o reducir considerablemente el tiempo que estarán activa las sesiones abiertas.
Dicha herramienta se ha utilizado en varias plataformas IPTV y en servicios de VoD, el mismo ha proporcionado un funcionamiento satisfactorio para cada caso. BCI, se ha comprometido con los clientes a ofrecer una gama de opciones de soporte, incluyendo reformulación de objetivos para aplicaciones específicas.

STB Virtual

Herramienta encargada para la ejecución de un componente de Middleware. Diseñado originalmente para simular el uso de un STB físico, incluyendo la emulación de las capacidades del hardware, del receptor físico para procesar el contenido que envía el proveedor, recibiendo mas de una conexión para la presentación de los contenidos IPTV en un monitor.

1.5. Conclusiones

En el presente capítulo se realiza un estudio de los sistemas existentes a nivel internacional, los cuales presentan arquitecturas similares enfocadas al campo de acción, aportando éstas un gran conocimiento y así colaborar de forma óptima con la arquitectura a proponer. Además se analizan las diferentes tendencias, tecnologías y metodologías actuales que podrían dar solución al problemas planteado.

CAPÍTULO 2: ARQUITECTURA DE SOFTWARE.

2.1. Introducción.

La arquitectura de software tiene una responsabilidad directa en el éxito de un sistema en cuestión. En este capítulo se muestra una descripción de los elementos fundamentales de una arquitectura base para los Set Top Box. Por tanto se hace necesario conocer y entender cuál es el significado de una arquitectura de dominio específico para utilizarlo como basamento en el desarrollo de este trabajo.

2.2. Arquitectura de Software.

Una Arquitectura de Software, también denominada Arquitectura lógica, consiste en un conjunto de patrones y abstracciones coherentes que proporcionan el marco de referencia necesario para guiar la construcción del software para un sistema de información.

La arquitectura de software establece los fundamentos para que analistas, diseñadores, programadores y demás miembros de un equipo de desarrollo trabajen en una línea común que permita alcanzar los objetivos del sistema de información, cubriendo todas las necesidades.

La arquitectura de software define, de manera abstracta, los componentes que llevan a cabo alguna tarea de computación, sus interfaces y la comunicación entre ellos.

La definición oficial de arquitectura de software que da la IEEE 1471-2000, plantea: La arquitectura de software es la organización fundamental de un sistema formada por sus componentes[1], las relaciones entre ellos y el contexto en el que se implantarán y los principios que sustentan su diseño y evolución. [5]

También Rumbaugh, Jacobson y Booch en su libro “El Proceso Unificado de Desarrollo de Software” dan una definición sobre la arquitectura de software y plantean que es el conjunto de decisiones significativas acerca de la organización de un sistema software, la selección de los elementos estructurales a partir de los cuales se compone el sistema y las interfaces entre ellos, junto con su comportamiento, tal y como se especifica en las colaboraciones entre esos elementos, la composición de estos elementos estructurales y de comportamiento en subsistemas progresivamente mayores, y el estilo arquitectónico que guía esta organización (…) [6]

[1] Bloques de construcción que conforman las partes de un sistema de software. A nivel de lenguaje de programación, pueden ser representados como módulos, clases, objetos o un conjunto de funciones relacionadas.

La práctica ha demostrado que resulta importante extender el concepto de arquitectura de software considerando los requerimientos y restricciones del sistema. La arquitectura de software puede considerarse entonces como el "puente" entre los requerimientos del sistema y la implementación. [7]
La arquitectura de software tiene como objetivo primario aportar elementos para ayudar a la toma de decisiones y proporciona conceptos y un lenguaje común que permiten la comunicación entre todos los que tienen parte en el proceso del desarrollo del software, también describe diversos aspectos del software de una forma comprensible utilizando modelos o vistas.
Los modelos o vistas de una arquitectura pueden expresarse mediante uno o varios lenguajes, el más obvio es el lenguaje natural, pero existen otros lenguajes, los cuales son apropiados únicamente para el modelado, dentro de estos últimos se encuentra el Lenguaje Unificado de Modelado (en lo adelante UML), el mismo ha sido adoptado de forma extendida para la mayoría de los modelos. Con el surgimiento de un nuevo proyecto para hacer un software no se crea una nueva arquitectura, lo que se realiza es la adopción de una arquitectura conocida valorando sus ventajas e inconvenientes frente al nuevo problema planteado. En caso de tener acumulada cierta experiencia en la construcción de software, entonces según los resultados obtenidos anteriormente, se podrá definir cual es la mejor arquitectura de software a aplicar en cada caso.

Las arquitecturas de software más conocidas son:

- ✓ **Monolítica:** donde el software se estructura en grupos funcionales muy acoplados.
- ✓ **Cliente-servidor:** donde el software reparte su carga de cálculos en dos partes independientes pero sin reparto claro de funciones.
- ✓ **Arquitectura por niveles o por capas:** esta constituye una especialización de la arquitectura cliente-servidor, donde la carga se distribuye en partes con un reparto claro de las funciones y donde cada capa tiene relación con la siguiente.
- ✓ **Arquitectura orientada a servicios:** Define la utilización de servicios para dar soporte a los requisitos del negocio.

La arquitectura de software representa las bases sobre las que se desarrollará el sistema, brinda una estructura que soporta las soluciones a cada tipo de problema que pueda aparecer en el desarrollo, asegura que los requerimientos más importantes puedan ser evaluados e implementados, permite flexibilidad en el sistema pues facilita la ejecución de futuros cambios y promueve la reutilización de componentes existentes como librerías de clases, sistemas legados y de aplicaciones de terceros.

La arquitectura de software provee una descripción de alto nivel de los subsistemas que comprenden la arquitectura del sistema. Está atada a comprender cómo las mayores operaciones del negocio son soportadas. Ayuda a la planificación de recursos y la asignación de tareas, debido a que el trabajo de desarrollo puede ser particionado a través de los subsistemas y los esfuerzos de desarrollo individual pueden proceder en paralelo. Cuando los equipos de desarrollo están geográficamente dispersados puede minimizar el número de interacciones requeridas.

Puede observarse que al hablar de arquitectura de software, de forma general, se hace alusión a la especificación de la estructura del sistema, entendida como la organización de componentes y relaciones entre ellos; los requerimientos que debe satisfacer el sistema y las restricciones a las que está sujeto, así como las propiedades no funcionales del sistema y su impacto sobre la calidad del mismo; las reglas y decisiones de diseño que gobiernan esta estructura y los argumentos que justifican las decisiones tomadas. [8]

2.2.1. Patrones de Arquitectura.

Se define al patrón como una regla que consta de tres partes, la cual expresa una relación entre un contexto, un problema y una solución. En líneas generales, un patrón sigue el siguiente esquema: [10]

- ✓ Contexto. Es una situación de diseño en la que aparece un problema de diseño.
- ✓ Problema. Es un conjunto de fuerzas que aparecen repetidamente en el contexto.
- ✓ Solución. Es una configuración que equilibra estas fuerzas. Esta abarca:
  - Estructura con componentes y relaciones
  - Comportamiento a tiempo de ejecución: aspectos dinámicos de la solución, como la colaboración entre componentes, la comunicación entre ellos, etc.

Los patrones de arquitectura expresan el esquema fundamental de organización para sistemas de software. Proveen un conjunto de subsistemas predefinidos; especifican sus responsabilidades e incluyen reglas y guías para organizar las relaciones entre ellos. Ayudan a especificar la estructura fundamental de una aplicación. Cada actividad de desarrollo es gobernada por esta estructura; por ejemplo, el diseño detallado de los subsistemas, la comunicación y colaboración entre diferentes partes del sistema. Cada patrón de arquitectura ayuda a conseguir una propiedad específica en el sistema global. Los patrones que dan soporte a características similares se agrupan en una misma categoría.

2.2.2. Estilos Arquitectónicos

El estilo arquitectónico constituye uno de los conceptos más importantes dentro de la arquitectura de software, describe y proporciona las propiedades básicas de una arquitectura y es encargado de imponer los límites de su evolución.

Cuando se define la arquitectura de software hay que decidir cual o cuales estilos arquitectónicos serán utilizados en la misma. La aplicación de dichos estilos mejora o disminuye la satisfacción de los atributos de calidad del sistema, es decir, la aplicación de un estilo depende en gran medida de los requisitos del sistema y de como se dará respuesta a los mismos.

En algunas bibliografías se puede encontrar que se utilice el término de estilo arquitectónico para referirse a los patrones o viceversa, pero por lo general se estudian de manera separada. Los dos términos se refieren a la estructura y la organización del sistema y la relación entre los elementos del mismo, por lo que la diferencia se aprecia en el nivel de abstracción en que se aplican. Los estilos arquitectónicos están a un nivel de abstracción más alto, mientras lo patrones dentro de ellos a un nivel mas cercano al diseño.

Según Pressman: “Cada estilo describe una categoría del sistema que contiene: un conjunto de componentes por ejemplo, una base de datos, módulos computacionales que realizan una función requerida por el sistema; un conjunto de conectores que posibilitan la comunicación, la coordinación y la cooperación entre los componentes; restricciones que definen como se pueden integrar los componentes que forman el sistema; y modelos semánticos que permiten al diseñador entender las propiedades globales de un sistema para analizar las propiedades conocidas de sus partes constituyentes”.[9]

Los estilos arquitectónicos no son más que arquitecturas comunes, marcos de referencias arquitectónicas o formas comunes que pueden ser aplicadas. Definen los posibles patrones que van a tener las aplicaciones y permiten evaluar las arquitecturas alternativas con ventajas y desventajas conocidas ante diferentes conjuntos de requerimientos no funcionales. Los estilos arquitectónicos comprenden los componentes (elementos), conectores, configuraciones y restricciones de las aplicaciones, así como sus relaciones y comportamiento, representando un alto nivel de abstracción.

Se define como estilo arquitectónico:
"Una familia de sistemas de software en términos de su organización estructural. Expresa componentes y las relaciones entre estos, con las restricciones de su aplicación y la composición asociada, así como también las reglas para su construcción. Así mismo, se considera como un tipo particular de estructura fundamental para un sistema de software, conjuntamente con un método asociado que especifica cómo construirlo. Éste incluye información acerca de cuándo usar la arquitectura que describe, sus invariantes y especializaciones, así como las consecuencias de su aplicación." [10]

Existe una gran variedad de estilos arquitectónicos las cuales tienen diversas clasificaciones entre las que se encuentran:

- ✓ Estilos de flujo de datos

  Esta familia de estilos enfatiza la reutilización y la capacidad de modificación. Es ideal para sistemas que realizan transformaciones de datos dividiéndolos en pasos sucesivos. En ella nos encontramos con dos tipos de componentes, el primero son los que realizan las transformaciones a los datos cuando pasan por ellos y el otro es el que se encarga de enviar el flujo de los datos desde la salida de un componente transformador a la entrada de otro, es decir, realiza el rol de comunicador.

  - Tuberías y filtro.

- ✓ Estilos centrados en datos.

Esta familia de estilos enfatiza la integrabilidad de los datos. Su uso es apropiado en sistemas que se centran en el acceso y actualización de datos en estructuras de almacenamiento que son compartidos por un número indefinido de componentes consumidores.

- Arquitectura de Pizarra o Repositorio.

✓ Estilos de llamada y retorno.

Esta familia de estilos enfatiza la capacidad de modificación y la escalabilidad. Son los estilos más generalizados en sistemas de gran escala. Miembros de la familia son las arquitecturas de programa principal y subrutina, los sistemas basados en llamadas a procedimientos remotos, los sistemas orientados a objetos y los sistemas jerárquicos en capas [10].

- Modelo-Vista-Controlador.
- Arquitectura en Capas.
- Arquitectura Orientada a Objetos.
- Arquitectura basada en Componentes.

✓ Estilos de código móvil.

Esta familia se caracteriza por su enorme portabilidad. Las arquitecturas que siguen este estilo tienen una parte del sistema que pertenece al propio entorno nativo de la máquina que lo incluye, la otra parte no. Ejemplos de la misma son los intérpretes, los sistemas basados en reglas y los procesadores de lenguaje de comando.

- Arquitectura de Máquinas Virtuales.

✓ Estilos heterogéneos.

Es la familia de estilos más fuertemente referida en los últimos tiempos, se incluyen en este grupo formas compuestas o indóciles a la clasificación en las categorías habituales. Podrían agregarse formas que aparecen esporádicamente en los estudios de nuevos estilos, como los sistemas de control de procesos industriales, sistemas de transición de estados, arquitecturas específicas de dominios o estilos derivados de otros estilos.

- Sistema de control de Procesos.
- Arquitecturas Basadas en Atributos.

✓ Estilos peer-to-peer.

Esta familia, también llamada de componentes independientes, enfatiza la modificabilidad por medio de la separación de las diversas partes que intervienen en la computación. Consiste por lo general en procesos independientes o entidades que se comunican a través de mensajes. Cada entidad puede enviar mensajes a otras entidades, pero no controlarlas directamente. Los mensajes pueden ser enviados a componentes nominados o emitidos mediante difusión [10].

- Arquitecturas basadas en Eventos.
- Arquitecturas Orientadas a Servicios (SOA).
- Arquitecturas basadas en Recursos.

2.2.3. Estilos arquitectónicos más importantes a analizar para su posible utilización.

2.2.3.1. Modelo-Vista-Controlador.

Reconocido como estilo arquitectónico por Taylor y Medvidovic. [13] Este se utiliza principalmente cuando es necesario tener de manera modular las interface de usuario, las reglas de negocio y el control de eventos.

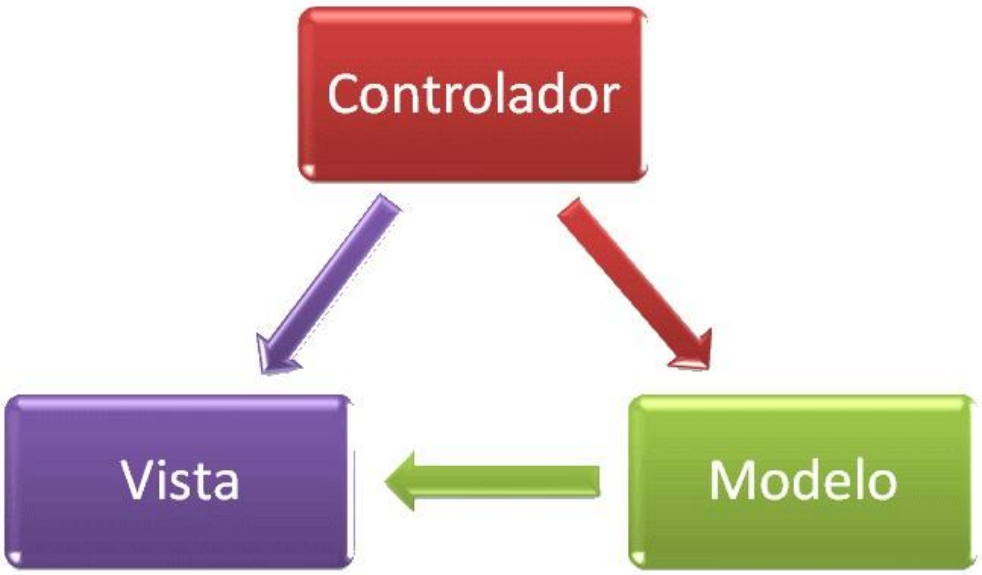


Figura.3 Estilo Modelo-Vista-Controlador

Modelo:

- ✓ Administra el comportamiento y los datos del dominio de aplicación, responde a requerimientos de información sobre su estado (usualmente formulados desde la vista y responde a instrucciones de cambiar el estado habitualmente desde el controlador). Mantiene el conocimiento del sistema. No depende de ninguna vista y de ningún controlador.

Vista:

- ✓ Maneja la Visualización de la información.

Controlador:

- ✓ Interpreta las acciones del ratón y el teclado, informando al modelo y/o a la vista para que cambien según resulte apropiado.

Ventajas:

- ✓ Permite el soporte de vistas múltiples.
- ✓ Permite la adaptación a los cambios, ya que al agregar nuevos tipos de vistas no afecta el modelo.
- ✓ Evita poner el código indebido en la capa impropia. Facilita el despliegue en caso de modificaciones en el modelo de datos.

Desventajas:

- ✓ Puede aumentar un poco la complejidad de la solución. Como está guiado por eventos puede ser algo más difícil de depurar.
- ✓ Si el modelo experimenta cambios frecuentes, por ejemplo podría desbordar las vistas con una lluvia de requerimientos de actualización.

2.2.3.2. Arquitectura Basada en Objetos.

En la bibliografía referente a estos estilos arquitectónicos se puede encontrar con nombres como Arquitecturas Basadas en Objetos, Abstracción de Datos y Organización Orientada a Objetos y perteneciente a la familia arquitectónica Llamada y Retorno. Los componentes de este estilo son los objetos, o más bien instancias de los tipos de datos abstractos.

Según David Garlan y Mary Shaw, los objetos representan una clase de componentes que ellos llaman *managers*, debido a que son responsables de preservar la integridad de su propia representación.

En la arquitectura Basada en Objeto sus componentes se basan en los principios de la programación orientada a objetos: encapsulamiento, herencia y polimorfismo. Son así mismo las unidades de modelado, diseño e implementación, y los objetos y sus interacciones son el centro de las incumbencias en el diseño de la arquitectura y en la estructura de la aplicación. Además las clases interfaces están separadas de sus implementaciones. En general la distribución de objetos es transparente.

Ventajas:

- ✓ Se puede modificar la implementación de un objeto sin afectar a sus clientes.
- ✓ Es posible descomponer problemas en colecciones de agentes de interacción.
- ✓ Un objeto es ante todo una entidad reutilizable en el entorno de desarrollo.

Desventajas:

- ✓ Para poder interactuar con otro objeto a través de una invocación de procedimiento, se debe conocer su identidad.
- ✓ Dependencia en cascada.
- ✓ Si se cambia un objeto se deben modificar todos los objetos o métodos que lo invocan.

2.2.3.3. Arquitecturas en Capas.

Las arquitecturas en capas constituyen uno de los estilos más conocidos y utilizados en la actualidad. Este estilo proporciona una organización jerárquica de tal manera que cada capa brinda servicios a la capa inmediatamente superior y se sirve de las prestaciones que le brinda la inmediatamente inferior. Cada capa suele ser una entidad compleja, formada por un conjunto de paquetes o subsistemas.

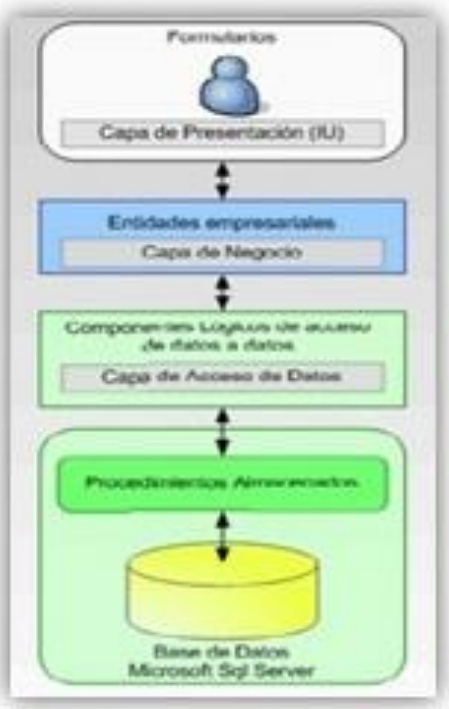


Figura. 4 Estilo en Capa

Ventajas:

- ✓ Soporta un diseño basado en niveles de abstracción creciente, lo cual permite la partición de un problema complejo en una secuencia de pasos incrementales.
- ✓ Proporciona amplia reutilización, ya que se pueden utilizar diferentes implementaciones o versiones de una misma capa en la medida que soporten las mismas interfaces de cara a las capas adyacentes.
- ✓ Mejora soporte del sistema al admitir muy naturalmente optimizaciones y refinamientos.

Desventajas:

- ✓ Puede ser difícil que componentes ubicar en cada una de las capas.
- ✓ Los cambios en las capas inferiores tienden a filtrarse hacia las superiores.

2.2.3.4. Arquitectura Orientada a Servicios.

La Arquitectura Orientada a Servicios (en lo adelante SOA), es un concepto de arquitectura de software que define la utilización de servicios para dar soporte a los requerimientos de software de usuario.

El concepto SOA hace referencia a un enfoque de arquitectura cuyo objetivo es la creación de sistemas a partir de servicios autónomos. Con SOA, la integración pasa a ser una reflexión previa más que una idea posterior. Probablemente, la solución final esté formada por servicios desarrollados en diferentes lenguajes de programación y se aloje en plataformas diferentes con numerosos modelos de seguridad y procesos empresariales [11].

Según IBM, este estilo es: “Una arquitectura de aplicación en la cual todas las funciones se definen como servicios independientes con interfaces bien definidas, que pueden ser llamadas en secuencias definidas para formar procesos de negocio”

SOA permite la creación de sistemas altamente escalables que reflejan el negocio de la organización, a su vez brinda una forma estándar de exposición e invocación de servicios, lo cual facilita la interacción diferentes sistemas propios o de terceros.

2.3. Lenguaje de descripción arquitectónica (ADL).

Desde el surgimiento de la disciplina arquitectura de software a principios de los 90 y en lo que va del siglo se han materializado diversas propuestas para describir y razonar en términos de arquitectura de software; muchas de ellas han asumido la forma de lenguajes de descripción arquitectónicos (en lo adelante ADL). Los ADL se utilizan para satisfacer requerimientos descriptivos de alto nivel de abstracción permitiendo descomponer un sistema en componentes y conectores, especificando de qué manera estos elementos se combinan para formar configuraciones y definiendo familias de arquitecturas o estilos.

Schneider define un ADL como una notación que permite una descripción y análisis preciso de las propiedades observables de una arquitectura de software, dando soporte a distintos estilos arquitectónicos a diferentes niveles de abstracción [12].

La delimitación categórica de los ADL es problemática, ocasionalmente otras notaciones se utilizan como si fueran ADL para las descripción de arquitecturas, un ejemplo lo constituye el UML, ha provocado que los ADL no ocupen el lugar que deberían.

Según Shaw y Garlan todo el ADL debe tener las siguientes propiedades [13]:

Composición:

- ✓ Permiten dividir un sistema complejo en partes más pequeñas, de manera jerárquica, o construir un sistema a partir de los elementos que lo constituyen.

Abstracción:

- ✓ Permiten identificar los distintos elementos en una estructura de alto nivel, así como su papel en la misma.

Reutilización:

- ✓ Permiten la reutilización tanto de los componentes y conectores como de la propia arquitectura.

Configuración:

- ✓ Permiten separar con claridad la descripción de los elementos individuales y de las estructuras - elementos compuestos en las que éstos participen.

Heterogeneidad:

- ✓ El ADL ha de ser independiente del lenguaje en que se implemente cada uno de los componentes que manipula; y debe diseñarse de modo que pueda combinar patrones arquitectónicos diferentes en un único sistema complejo.

Flexibilidad:

- ✓ Permiten la definición de nuevas formas de interacción entre componentes.

Análisis:

- ✓ Permiten diversas formas de análisis de la arquitectura, de modo que se puedan determinar sus propiedades con independencia de una implementación concreta, así como verificarlas después de cualquier modificación.

Entre los principales ADL que se usan en la actualidad se encuentran:

Acme:

- ✓ Se define como una herramienta capaz de soportar el mapeo de especificaciones arquitectónicas entre diferentes ADL, en otras palabras, como un lenguaje de intercambio de arquitectura. No es entonces un ADL en sentido estricto, aunque la literatura de referencia acostumbra tratarlo como tal. De hecho, posee numerosas prestaciones que también son propias de los ADL [14] y los desarrollos actuales profundizan su capacidad intrínseca como ADL puro.

Armani:

- ✓ Es un lenguaje puramente declarativo que describe la estructura del sistema y las restricciones a respetar, pero no hace referencia alguna a la generación del sistema o a la verificación de sus propiedades no funcionales. Se basa en siete entidades para describir las instancias del diseño: componentes, conectores, puertos, roles, sistemas, representaciones y propiedades [14].

Aesop:

- ✓ El nombre oficial *es Aesop Software Architecture Design Environment Generator* y se centra en la especificación propiamente dicha de arquitecturas. Basado en componentes, con un sistema de tipos extensible donde los puntos de interfaz se modelan como puertos de entrada y salida. La evolución la permite mediante un sub-tipado que preserva el comportamiento. Respecto a las propiedades no funcionales, solo permite que se asocien textos arbitrarios a los componentes.

Jacal:

- ✓ Es un lenguaje de descripción de arquitecturas de software de propósito general. Reúne ventajas de otros lenguajes existentes y tiene la virtud de permitir la ejecución (animación) de las arquitecturas descritas. Mediante este procedimiento se puede comprobar o refutar propiedades deseables de los diseños y recopilar métricas dinámicas. Además cuenta con una representación gráfica que permita a simple vista transmitir la arquitectura del sistema, sin necesidad de recurrir a información adicional.

Darwin:

- ✓ Es un lenguaje de descripción arquitectónica desarrollado por Jeff Magee y Jeff Kramer. El mismo describe un tipo de componente mediante una interfaz consistente en una colección de servicios que son ya sea provistos (declarados por ese componente) o requeridos (o sea, que se espera ocurran en el entorno) [14]. Darwin no proporciona una base adecuada para el análisis de la conducta de una arquitectura, debido a que el modelo no dispone de ningún medio para describir las propiedades de un componente o de sus servicios más que como comentario [15].

2.4. Lenguaje Unificado de Modelado (UML).

UML es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema software orientado a objetos.

Un modelo UML está compuesto por tres clases de bloques de construcción (ver Anexo 4: Componentes UML):

- ✓ **Elementos**:
  Son abstracciones de cosas reales o ficticias (estructurales, comportamiento, agrupación, anotación).
- ✓ **Relaciones:**
  Relaciones entre los elementos (dependencia, asociación, generalización).

- ✓ **Diagramas:**
  Son los objetos con sus relaciones (clases, objetos, casos de uso, secuencia, colaboración, estados, actividades, componentes, despliegue).

Un sistema de software es representado por este lenguaje a través de cinco vistas o modelos principales separados, pero relacionados entre sí, denominadas vista de caso de uso, lógica, de implementación, de procesos y de despliegue. Cada uno de estas vistas utiliza diversos diagramas para describir el sistema, los cuales se dividen en estáticos y dinámicos.

UML no es considerado un lenguaje de descripción arquitectónica ya que carece del concepto de estilo y su forma de expresar ciertas características, sobre todo las dinámicas no son suficientes para los arquitectos. A pesar de esto, UML es el lenguaje más usado en el modelado de sistemas de software, el cual describe a la arquitectura de software mediante las cinco vistas mencionadas anteriormente y cuenta con algunos conceptos utilizados por esta como interfaz, componentes, etc.

2.5. Marcos de trabajo (*Frameworks*).

Dentro del desarrollo de software un marco de trabajo o framework, por su término en inglés, no es más que una estructura de soporte definida por la cual un proyecto de software puede ser organizado y desarrollado. Define una arquitectura genérica, compuesta por una serie de componentes personalizables y sus interfaces, así como las reglas y mecanismos de integración entre ellos.

Los framework están basados en la idea de permitir la producción fácil de sistemas de software pertenecientes a un dominio específico utilizando una estructura genérica.

Los objetivos principales que persigue un framework son:

- ✓ El desarrollo rápido de aplicaciones.
- ✓ La reutilización de componentes software.
- ✓ Promover buenas prácticas de desarrollo como el uso de patrones.

2.6. Herramienta asistida por computadora para el modelado Visual Paradigm.

Visual Paradigm es una herramienta para un diseño profesional, que soporta el ciclo de vida completo del desarrollo de software: análisis y diseño orientados a objetos, construcción, pruebas y despliegue. El Lenguaje de Modelado Unificado ayuda a una rápida construcción de aplicaciones de calidad, con un menor costo. Permite dibujar todos los tipos de diagramas de clases, código inverso, generar código desde diagramas y generar documentación. La herramienta también proporciona abundantes tutoriales de UML, demostraciones interactivas de UML y proyectos UML.

Características de Visual Paradigm:

- ✓ Soporte de UML versión 2.1.
- ✓ Diagramas de Procesos de Negocio: Proceso, Decisión, Actor de negocio, Documentos.
- ✓ Modelado colaborativo con CVS *y Subversion* (nueva característica) interoperabilidad con modelos UML2 (meta modelos UML 2.x para plataforma Eclipse) a través de XML (nueva característica).
- ✓ Ingeniería inversa: Código a modelo, código a diagrama.
- ✓ Ingeniería inversa Java, C++, Esquemas XML, XML, .NET (.exe / .dll), CORBA IDL.
- ✓ Generación de código: Modelo a código, diagrama a código.
- ✓ Editor de Detalles de Casos de Uso: Entorno todo en uno para la especificación de los detalles de los casos de uso, incluyendo la especificación del modelo general y de las descripciones de los casos de uso.
- ✓ Diagramas EJB - Visualización de sistemas EJB.
- ✓ Generación de código y despliegue de EJB´s: Generación de *beans* para el desarrollo y despliegue de aplicaciones.
- ✓ Diagramas de flujo de datos.
- ✓ Soporte ORM: Generación de objetos Java desde la base de datos.
- ✓ Generación de bases de datos: Transformación de diagramas de Entidad-Relación en tablas de base de datos.

- ✓ Ingeniería inversa de bases de datos: Desde Sistemas Gestores de Bases de Datos (DBMS) existentes a diagramas de Entidad-Relación.
- ✓ Generador de informes para generación de documentación Distribución automática de diagramas - Reorganización de las figuras y conectores de los diagramas UML.
- ✓ Importación y exportación de ficheros XML.
- ✓ Integración con Visio: Dibujo de diagramas UML con plantillas (*stencils*) de MS Visio.
- ✓ Editor de figuras.

2.7. Metodologías del desarrollo del Software.

Cuando se va a comenzar con el desarrollo de un software algo fundamental es tener seleccionada la Metodología de Desarrollo que se va a utilizar. Lo más recomendable es realizar un estudio de las metodologías más utilizadas dentro del proceso de desarrollo de software, para así tener un conocimiento base y realizar una buena selección.

2.7.1. Extreme Programming (XP).

Es una de las metodologías de desarrollo de software más exitosas y utilizadas en la actualidad para el desarrollo de proyectos de corto plazo y corto equipo. La metodología consiste en una programación rápida o extrema, cuya particularidad es tener como parte del equipo, al usuario final, pues es uno de los requisitos para llegar al éxito del proyecto.

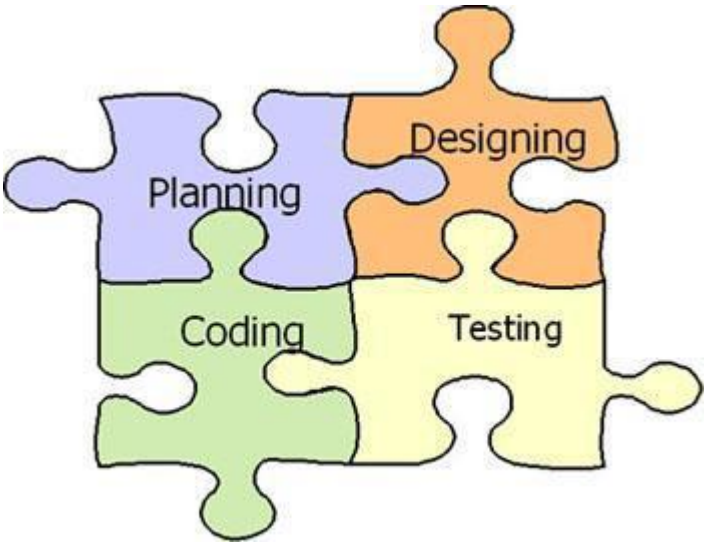


Figura.5 Metodología Extreme Programming.

Esta metodología se basa en:

- ✓ **Pruebas Unitarias:** se basa en las pruebas realizadas a los principales procesos, de tal manera que adelantándonos en algo hacia el futuro, podamos hacer pruebas de las fallas que pudieran ocurrir. Es como si nos adelantáramos a obtener los posibles errores.

- ✓ **Re-fabricación:** se basa en la reutilización de código, para lo cual se crean patrones o modelos estándares, siendo más flexible al cambio.

- ✓ **Programación en pares:** una particularidad de esta metodología es que propone la programación en pares, la cual consiste en que dos desarrolladores participen en un proyecto en una misma estación de trabajo. Cada miembro lleva a cabo la acción que el otro no está haciendo en ese momento.

Lo fundamental en este tipo de metodología es:

- ✓ La comunicación, entre los usuarios y los desarrolladores.

- ✓ La simplicidad, al desarrollar y codificar los módulos del sistema.
- ✓ La retroalimentación, concreta y frecuente del equipo de desarrollo, el cliente y los usuarios finales.

2.7.2. Microsoft Solution Frameworks (MSF).

El MSF es una metodología flexible e interrelacionada con una serie de conceptos, modelos y prácticas de uso, que controlan la planificación, el desarrollo y la gestión de proyectos. MSF se centra en los modelos de proceso y de equipo dejando en un segundo plano las elecciones tecnológicas.

Figura.6 Metodología MSF

MSF tiene las siguientes características:

- ✓ **Adaptable:** es parecido a un compás, usado en cualquier parte como un mapa, del cual su uso es limitado a un específico lugar.
- ✓ **Escalable:** puede organizar equipos tan pequeños entre 3 o 4 personas, así como también, proyectos que requieren 50 personas a más.
- ✓ **Flexible:** es utilizada en el ambiente de desarrollo de cualquier cliente.
- ✓ **Tecnología Agnóstica:** porque puede ser usada para desarrollar soluciones basadas sobre cualquier tecnología.

MSF se compone de varios modelos encargados de planificar las diferentes partes implicadas en el desarrollo de un proyecto: Modelo de Arquitectura del Proyecto, Modelo de Equipo, Modelo de Proceso, Modelo de Gestión del Riesgo, Modelo de Diseño de Proceso y por último el Modelo de Aplicación.

- ✓ **Modelo de Arquitectura del Proyecto:** Diseñado para acortar la planificación del ciclo de vida. Este modelo define las pautas para construir proyectos empresariales a través del lanzamiento de versiones.

- ✓ **Modelo de Equipo:** Este modelo ha sido diseñado para mejorar el rendimiento del equipo de desarrollo. Proporciona una estructura flexible para organizar los equipos de un proyecto. Puede ser escalado dependiendo del tamaño del proyecto y del equipo de personas disponibles.

- ✓ **Modelo de Proceso:** Diseñado para mejorar el control del proyecto, minimizando el riesgo, y aumentar la calidad acortando el tiempo de entrega. Proporciona una estructura de pautas a seguir en el ciclo de vida del proyecto, describiendo las fases, las actividades, la liberación de versiones y explicando su relación con el Modelo de equipo.

- ✓ **Modelo de Gestión del Riesgo:** Diseñado para ayudar al equipo a identificar las prioridades, tomar las decisiones estratégicas correctas y controlar las emergencias que puedan surgir. Este modelo proporciona un entorno estructurado para la toma de decisiones y acciones valorando los riesgos que puedan provocar.

- ✓ **Modelo de Diseño del Proceso:** Diseñado para distinguir entre los objetivos empresariales y las necesidades del usuario. Proporciona un modelo centrado en el usuario para obtener un diseño eficiente y flexible a través de un enfoque iterativo. Las fases de diseño conceptual, lógico y físico proveen tres perspectivas diferentes para los tres tipos de roles: los usuarios, el equipo y los desarrolladores.

- ✓ **Modelo de Aplicación:** Diseñado para mejorar el desarrollo, el mantenimiento y el soporte, proporciona un modelo de tres niveles para diseñar y desarrollar aplicaciones software. Los

servicios utilizados en este modelo son escalables, y pueden ser usados en un solo ordenador o incluso en varios servidores.

2.7.3. Rational Unified Process (RUP).

La metodología RUP es un proceso de desarrollo de software y junto con el UML, constituye la metodología estándar más utilizada para el análisis, implementación y documentación de sistemas orientados a objetos.

El RUP no es un sistema con pasos firmemente establecidos, sino un conjunto de metodologías adaptables al contexto y necesidades de cada organización.

RUP se caracteriza por ser centrado en la arquitectura, ya que la organización del sistema depende de los casos de usos claves y debe tener en cuenta la comprensibilidad, la facilidad de adaptación al cambio y la reutilización. Los casos de uso claves son aquellos que más le interesan al cliente. Es un proceso iterativo e incremental, el trabajo se divide en partes llamadas iteraciones, en cada una de ellas se recorren todos los flujos que RUP propone (Modelo del negocio, Requerimiento, Análisis y Diseño, Implementación, etc.). Y por último RUP es dirigido por casos de uso, porque son estos los que especifican las funcionalidades con las que va a contar el sistema. Los casos de uso no sólo son una herramienta para especificar los requisitos del sistema, también guían su diseño, implementación y pruebas, es decir, guían todo el desarrollo software.

RUP divide el desarrollo del software en cuatro fases:

- ✓ **Inicio:** El Objetivo en esta etapa es determinar la visión del proyecto.
- ✓ **Elaboración:** En esta etapa el objetivo es determinar la arquitectura óptima.
- ✓ **Construcción:** En esta etapa el objetivo es llegar a obtener la capacidad operacional inicial.
- ✓ **Transición:** El objetivo es llegar a obtener el entregable del proyecto.

El ciclo de vida que se desarrolla por cada iteración es llevada bajo dos disciplinas:

Disciplina de desarrollo:

- ✓ **Modelo del Negocio:** Entendiendo las necesidades del negocio. Identifica quienes participan en el negocio y las actividades que requieren automatización.

- ✓ **Requerimientos:** Trasladando las necesidades del negocio a un sistema automatizado. Se definen las funcionalidades requeridas y las restricciones.

- ✓ **Análisis y Diseño:** Trasladando los requerimientos dentro de la arquitectura de software.

- ✓ **Implementación:** Creando software que se ajuste a la arquitectura y que tenga el comportamiento deseado.

- ✓ **Pruebas:** Asegurándose que el comportamiento requerido es el correcto y que todo lo solicitado está presente. Busca defectos lo largo del ciclo de vida del software.

- ✓ **Despliegue:** Produce entregable del producto y realiza actividades (empaque, instalación, asistencia a usuarios, etc.) para entregar el software a los usuarios finales.

Disciplina de soporte:

- ✓ **Gestión de configuración y cambio:** Guardando todas las versiones del proyecto. El control ayuda a evitar confusiones costosas.

- ✓ **Gestión de proyecto:** Proporcionar directrices prácticas para la planificación, selección de personal, ejecución y supervisión de los proyectos. Proporcionar una infraestructura para gestionar los riesgos.

- ✓ **Ambiente:** La disciplina de entorno proporciona el entorno de soporte para un proyecto. Proporciona a la empresa de desarrollo de software un entorno de desarrollo de software (los procesos y las herramientas) que den soporte al equipo de desarrollo.

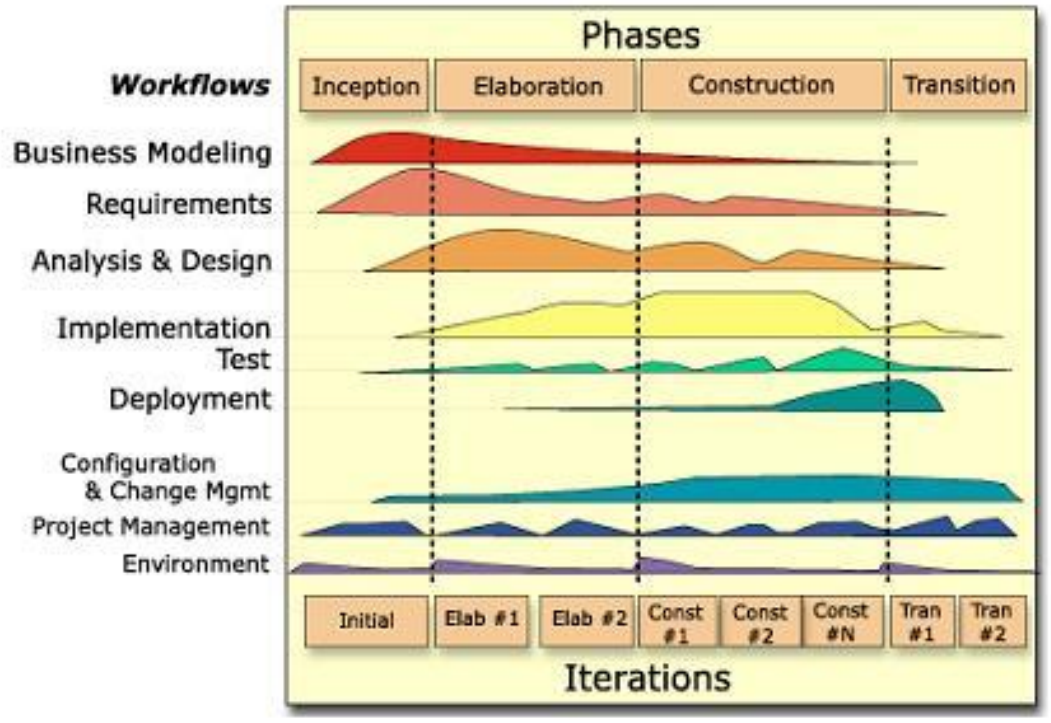


Figura.7 Fases e iteraciones de la metodología RUP.

Los elementos de RUP son:

- ✓ **Actividades:** Son los procesos que se llegan a determinar en cada iteración.
- ✓ **Trabajadores:** Vienen hacer las personas o entes involucrados en cada proceso.
- ✓ **Artefactos:** Un artefacto puede ser un documento, un modelo, o un elemento de modelo.

Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software.

2.8. Arquitecto de Software.

El arquitecto de software es considerado hoy en día como un integrante fundamental en los proyectos de desarrollo de software. La IEEE define al arquitecto de software de esta manera: “es una persona, equipo u organización responsable por la arquitectura del sistema” [23].

El arquitecto de software es el que toma las principales decisiones acerca de cómo será construido el software por los programadores, por lo que debe dominar la mayor cantidad de tecnologías de software y prácticas de diseño para lograr garantizar el rehúso, robustez, portabilidad, flexibilidad, escalabilidad y mantenibilidad de las aplicaciones. Las decisiones que el arquitecto toma deberán ser plasmadas en una notación formal estandarizada como lo es UML, sobre todo si se utilizan las nuevas tecnologías, en especial con los lenguajes orientados a objetos.

El arquitecto de software debe contar con una serie de características y capacidades, las que RUP plantea de esta forma:

El arquitecto de software debe disponer de formación, madurez, visión y una amplia experiencia que permita recuperar cuestiones rápidamente y realizar valoraciones educadas y críticas en ausencia de la información completa. Más concretamente, el arquitecto de software, o los miembros del equipo de arquitectura, deben combinar estas habilidades [24].

- ✓ Experiencia en el dominio del problema, a través de una comprensión precisa de los requisitos, y el dominio de ingeniería de software. Si hay un equipo, estas cualidades pueden abarcar los diferentes miembros del equipo, pero como mínimo un arquitecto de software debe proporcionar la visión global para el proyecto [24].
- ✓ Liderazgo para dirigir el esfuerzo técnico entre los diferentes equipos, y para tomar decisiones críticas bajo presión y adherirse a estas decisiones. Para ser efectivos, el arquitecto de software y el gestor de proyectos deben trabajar conjuntamente, con el arquitecto de software dirigiendo las cuestiones técnicas y el gestor de proyectos dirigiendo las cuestiones administrativas. El arquitecto de software debe tener autoridad para tomar decisiones técnicas [24].

- ✓ Comunicación para ganar confianza, persuadir, motivar y orientar. El arquitecto de software no puede dirigir por decreto, sólo con el consentimiento del resto del proyecto. Para ser efectivos, el arquitecto de software debe ganar el respeto del equipo de proyecto, el gestor del proyecto, el cliente y la comunidad de usuarios, así como del equipo de gestión [24].
- ✓ Orientación a objetivos y productividad centrándose exclusivamente en los resultados. El arquitecto de software es la fuerza técnica dirigente detrás del proyecto, no un visionario o un soñador. La carrera de un arquitecto de software con éxito es una larga serie de decisiones sub-óptimas efectuadas en momentos de incertidumbre y bajo presión. Sólo quienes puedan centrarse en lo que se debe hacer tendrán éxito en este entorno del proyecto [24].

2.9. Arquitectura de software de dominio específico.

Un dominio es un área de interés, usualmente representando un espacio del problema que es un subconjunto de una o más disciplinas. [16]

Un dominio es definido por un conjunto de problemas o funciones comunes que las aplicaciones en ese dominio pueden resolver o hacer.

La idea básica de la arquitectura de software de dominio específico (en lo adelante DSSA) es hacer práctica la reutilización de software, enfocándose en los problemas específicos de dominios de software y desarrollando soluciones basadas en componentes para estos dominios de problemas. Estos componentes son genéricos y necesitan ser configurados en el momento que la aplicación es implementada.

DARPA[2] definió 4 pasos claves para las DSSA: [17]

- ✓ Un modelo del dominio es desarrollado para establecer una terminología estándar y una semántica común para el dominio del problema.

[2] Agencia de Investigación de Proyectos Avanzados de Defensa, es una agencia del departamento de defensa de los Estados Unidos responsable del desarrollo de nuevas tecnologías para uso militar.

- ✓ Un conjunto de requerimientos de referencia para todas las aplicaciones dentro del dominio del problema que es desarrollado.
- ✓ Una arquitectura de referencia es definida para una solución genérica, estandarizada y basada en componentes del sistema para anticipar los requerimientos del cliente.
- ✓ Nuevas aplicaciones son desarrolladas para determinar requerimientos específicos de la aplicación, seleccionar o generar componentes particulares y ensamblarlos acorde a la arquitectura de referencia.

La arquitectura de referencia puede incluir componentes concretos que deberían ser embebidos en todas las aplicaciones que derivan de ella. Además es adaptada para resolver los requerimientos particulares de los clientes.

La actividad principal es ajustar a la medida, la arquitectura de referencia, usando los requerimientos de la aplicación para producir la arquitectura específica para la aplicación.

2.10. Conclusiones.

En el presente capítulo se han expuesto los principales aspectos dentro de la disciplina arquitectura de software. Además se realizó un estudio detallado de los principales estilos y patrones arquitectónicos que permitían lograr un diseño de alto nivel lo más robusto y escalable posible, pero que a la vez se ajuste a las características propias de la arquitectura de los STB. Finalmente la decisión se inclinó al uso de una arquitectura en capas.

También se estudiaron otros aspectos como la metodología de desarrollo, lenguajes y otras herramientas que complementan la arquitectura de este sistema. Se definió UML como lenguaje de modelado para los artefactos propuestos. Se han identificado elementos importantes a tener en cuenta en las temáticas abordadas sobre la Ingeniería de Software, modelado de procesos de software, proceso, metodología y herramientas llegando a la conclusión de que no existe una opción que lo reúna todo, se considera que es posible combinarlas. Por ello la propuesta debe basarse en la metodología RUP y debe estar compuesto por el orden lógico de actividades, los roles, las tareas, los artefactos de entrada y salida. Esto garantiza el uso de buenas prácticas y el desarrollo de flujos de trabajo organizado, estructurado y disciplinado, donde la organización tenga capacidad para repetirlos y mejorarlos.

CAPÍTULO 3: PROPUESTA DE ARQUITECTURA.

3.1. Introducción.

En los capítulos anteriores se ha analizado y hecho referencia a una serie de elementos fundamentales que contribuyen en el proceso de descripción y creación de la propuesta de arquitectura deseada.

Con esta propuesta se pretende emular el funcionamiento de un STB en la plataforma IPTV. La solución permite acoplarse a una plataforma IPTV determinada y consumir los servicios que esta brinda. Dichos servicios se muestran al usuario con la funcionalidad y la calidad semejante a la de un STB convencional (*hardware*). La construcción de dicha propuesta se basa fundamentalmente en los requerimientos y contribuciones de la UIT para un terminal de acceso IPTV. La propuesta de arquitectura genérica para un STB IPTV de la UIT se definió como la línea base de la arquitectura a cometer.

3.2. Objetivos y restricciones arquitectónicas.

Los objetivos y restricciones de la arquitectura se determinan en gran medida por las funcionalidades básicas de la solución a desarrollar y los requisitos no funcionales, que no son más que las cualidades y propiedades que el producto debe tener.

3.2.1. Principales Funcionalidades.

- ✓ Gestión de contenidos en la plataforma IPTV.
- ✓ Gestión de perfiles de usuarios de la plataforma IPTV.
- ✓ Gestión de las características de la plataforma IPTV.
- ✓ Encriptación/Desencriptación de contenidos.
- ✓ Codificación/Decodificación, demodulación y desmultiplexación de los contenidos.
- ✓ Reproducción de los contenidos.

3.2.2. Requisitos no funcionales.

3.2.2.1. Requerimiento de Software.

- ✓ El emulador de STB deberá correr sobre los sistemas operativos Windows, GNU/Linux.

3.2.2.2. Requerimiento de Hardware.

- ✓ La PC donde se instale la aplicación debe tener los siguientes requerimientos mínimos: Microprocesador 1.7 GHz, 256 MB de RAM, Red Ethernet 100 Mbps, 10 GB de espacio en disco duro.

3.2.2.3. Requerimiento de Usabilidad.

- ✓ El sistema debe ser intuitivo y de alta usabilidad para los usuarios no expertos en informática. Debe permitir explotar al máximo las funcionalidades con poco tiempo de adiestramiento.

3.2.2.4. Requerimiento de Seguridad.

- ✓ El sistema debe gestionar la seguridad en la totalidad de las comunicaciones con la plataforma.

3.2.2.5. Requerimiento de Rendimiento.

- ✓ El tiempo de respuesta ante los eventos provocados por el usuario debe ser lo más breve posible, teniendo en cuenta que el sistema funciona a través de la gestión de peticiones a la plataforma así como de eventos locales.

3.2.2.6. Requerimiento de Soporte.

- ✓ Se le debe dar mantenimiento y soporte al software y las aplicaciones diseñadas para ser ejecutadas sobre el mismo.

3.2.2.7. Requerimiento de Confiabilidad.

- ✓ El sistema debe estar libre de fallas, chequear permanentemente la integridad de los datos así como la calidad de los servicios que brinda.

3.2.2.8. Requerimiento de Escalabilidad.

- ✓ La solución debe ser los mas genérica posible, de forma tal que sea receptiva a los cambios sin que varíe la esencia de las funcionalidades principales. Esto permite la fácil integración con otras aplicaciones y componentes que se deseen agregar en el proceso de integración.

3.3. Descripción de las vistas de la arquitectura.

Después de haber realizado un estudio de los diferentes estilos arquitectónicos, se decidió optar por el estilo “Llamada y Retorno”, enfatizando en la arquitectura en capas, debido a su capacidad de modificación y escalabilidad, así como la facilidad en el diseño de sistemas jerárquicos en capas y orientados a objetos.

La siguiente vista muestra el funcionamiento de un STB en la plataforma IPTV. En la misma podemos identificar una serie de componentes fundamentales de la propuesta a desarrollar, así como algunos flujos internos del STB y la interacción del mismo con la plataforma a través de los protocolos propuestos.

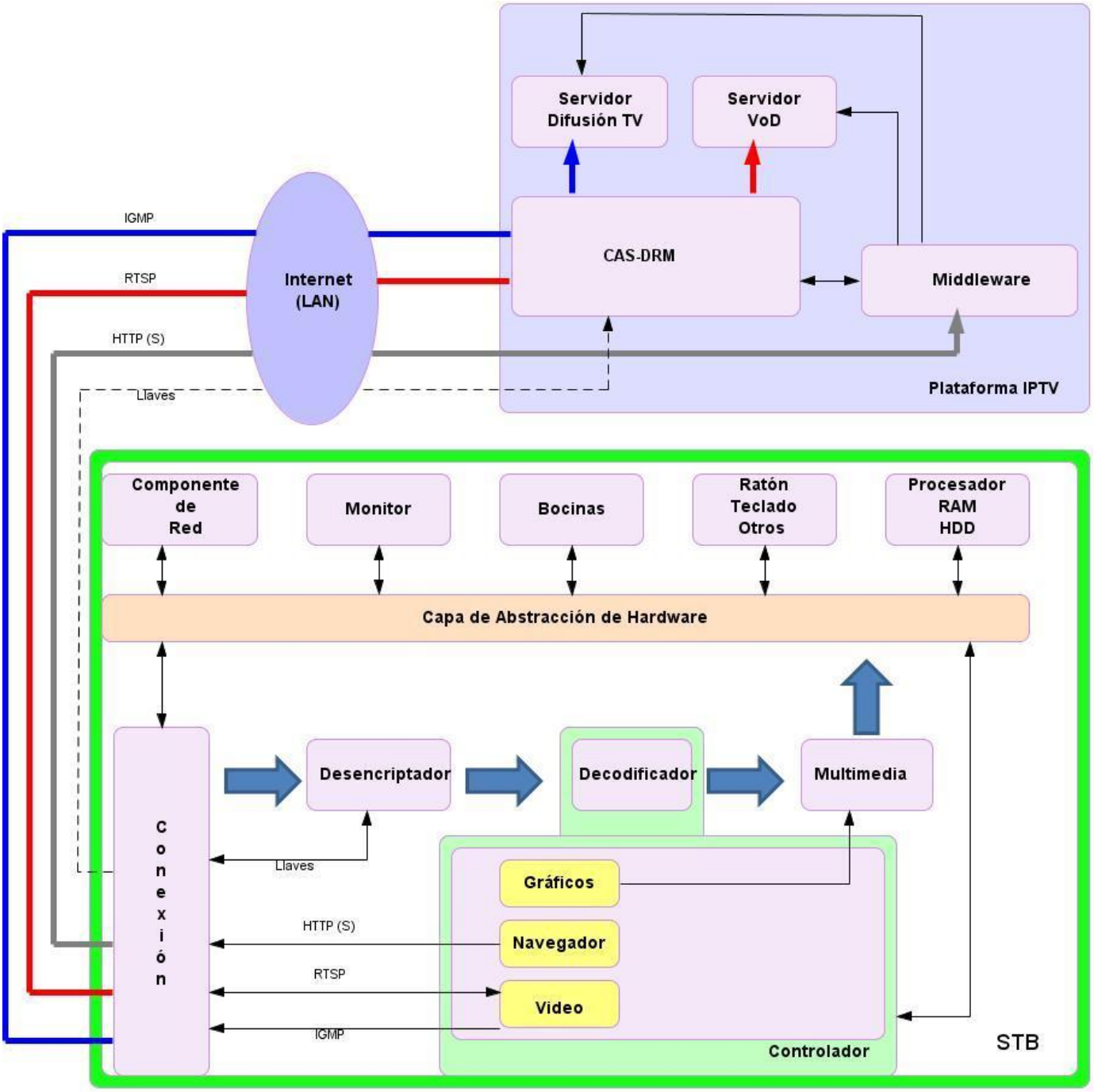


Figura.8 STB en la Plataforma IPTV.

3.4. Propuesta de Arquitectura.

La propuesta consiste en la definición de una serie de capas compuestas fundamentalmente por paquetes o subsistemas destinados a implementar las funcionalidades de un STB. Además se representa la relación entre las mismas determinada por el flujo de datos en la aplicación.

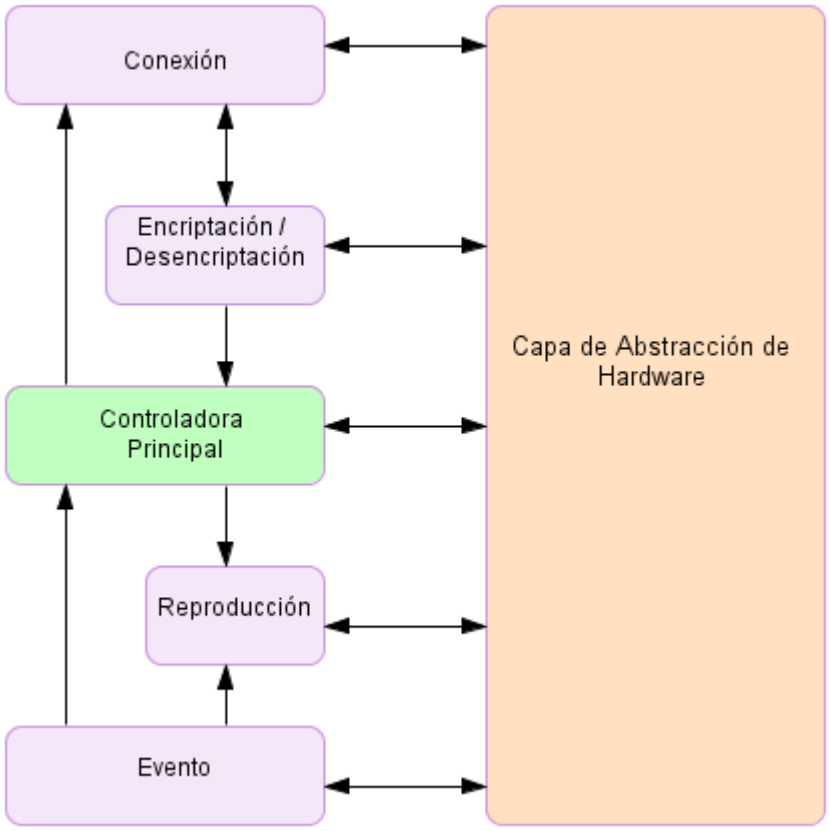


Figura.9 Vista principal de la arquitectura.

3.5. Capa Conexión

La capa de conexión se encarga fundamentalmente de la comunicación del STB con la plataforma IPTV, además de la recepción de datos y el envío de peticiones relacionados por el contenido.

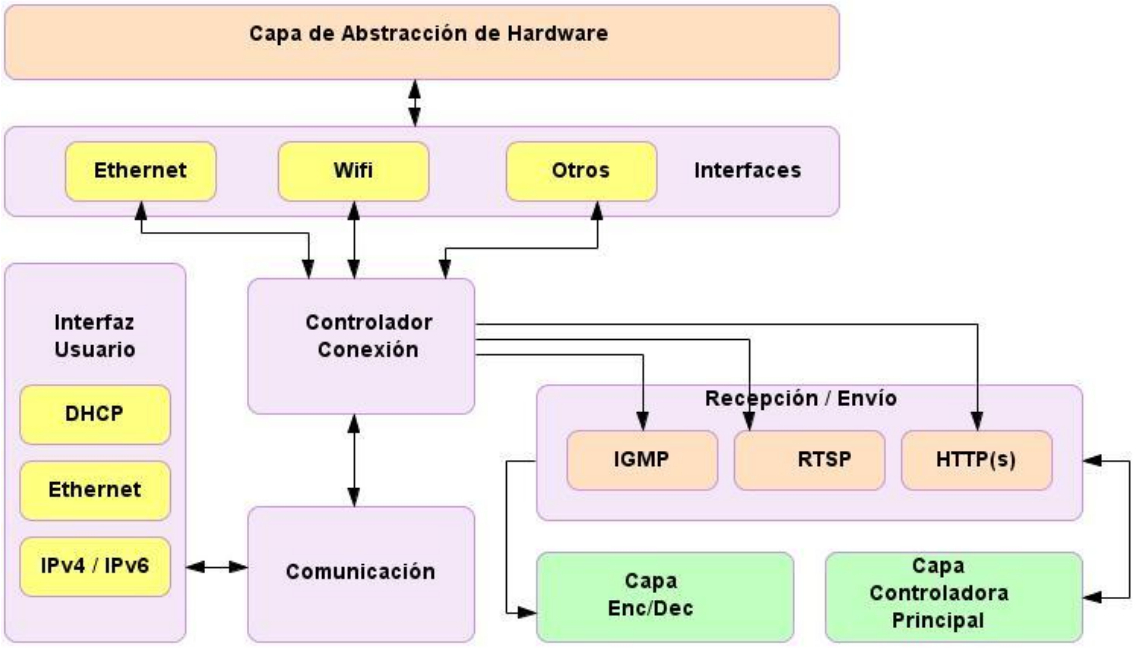


Figura.10 Capa Conexión.

Esta conformada por 3 componentes principales: controlador, recepción/envío y comunicación. Estos a su vez se comunican con otras capas a través de varias interfaces.

- ✓ **Controlador conexión.**

  Encargado de controlar y gestionar el funcionamiento de la capa de conexión, así como la comunicación con la Capa de Abstracción de Hardware (en lo adelante CAH). Esta comunicación se realiza mediante la Interfaz de Ethernet y la Interfaz Wi-Fi las cuales poseen funcionalidades necesarias para la comunicación con el puerto Ethernet y algún dispositivo de conexión Wi-Fi (en caso de contar con el mismo) respectivamente, usando la CAH como intermediaria hacia los dispositivos de hardware en todo momento.

- ✓ **Recepción/Envío**

La recepción y envío se divide en dos flujos fundamentales: Difusión de TV y VoD.

La difusión de TV se gestiona a través del protocolo IGMP por tanto esta capa de conexión y el componente en especial debe implementar las funcionalidades necesarias para establecer comunicaciones con la plataforma regidas por el protocolo antes mencionado.

Las funcionalidades de VoD en cambio rigen su comunicación a través de los protocolos de *streaming* de video en tiempo real, así como para el control de la calidad del servicio (QoS).

Además se propone gestionar la conexión y comunicación con la plataforma a través del protocolo HTTPS, orientado a transacciones, basado en el esquema petición-respuesta entre un cliente (STB) y el servidor (Plataforma IPTV) y diseñado para crear un canal cifrado apropiado para el tráfico de información sensible.

- ✓ **Comunicación**

Encargado de gestionar y establecer la comunicación entre el STB y la plataforma IPTV. Implementa fundamentalmente el uso del protocolo DHCP el cual permite al STB obtener y establecer parámetros de configuración automática o manualmente para integrarse a la plataforma a través del uso de direcciones IP, tanto en IPv4 como en IPv6. El emulador de STB en este caso puede utilizar la configuración de red existente en el ordenador donde esta instalado, además modificar parámetros de la misma a través de una *Interfaz de Configuración de Red.*

3.6. Capa Encriptación / Desencriptación.

La capa de encriptación/desencriptación agrupa las funcionalidades para desencriptar el contenido recibido de la plataforma, que por necesidades de seguridad y privacidad requiere de las llaves

necesarias. Estas llaves son asignadas al usuario según sus peticiones así como privilegios de contrato y son gestionadas en el STB de forma automática.

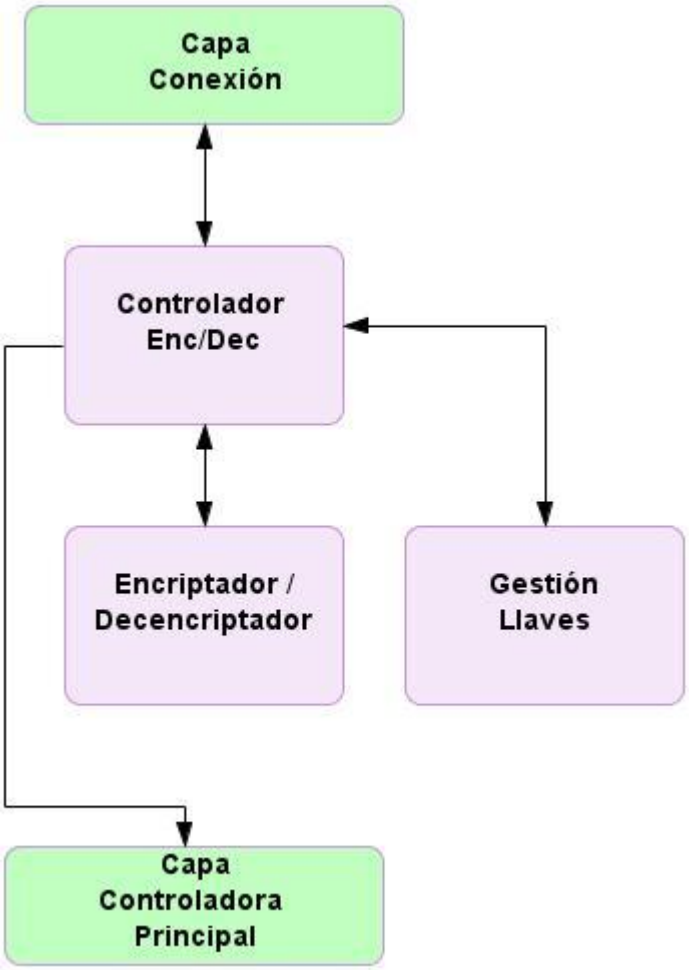


Figura.11 Capa Encriptación / Desencriptación.

Podemos dividir esta capa en tres elementos fundamentales:

- ✓ **Encriptador/Desencriptador.**

  Implementa las funcionalidades y algoritmos necesarios para encriptar y desencriptar el contenido y los datos en el STB.

- ✓ **Gestión de llaves.**

Maneja el proceso de obtención, aplicación, renovación y utilización de las llaves del STB gestionadas en la plataforma IPTV.

- ✓ **Controlador E/D.**

Controla y gestiona la encriptación y desencriptación de los contenidos recibidos de la plataforma, además de la entrega de los mismos a la Capa Controladora Principal.

3.7. Capa Controladora Principal.

Esta es la capa fundamental en la arquitectura propuesta, pues es la encargada de gestionar y procesar el contenido, así como programar y configurar el funcionamiento del STB y las aplicaciones del mismo. Como su nombre lo indica, es la principal controladora de los procesos y funcionalidades del STB.

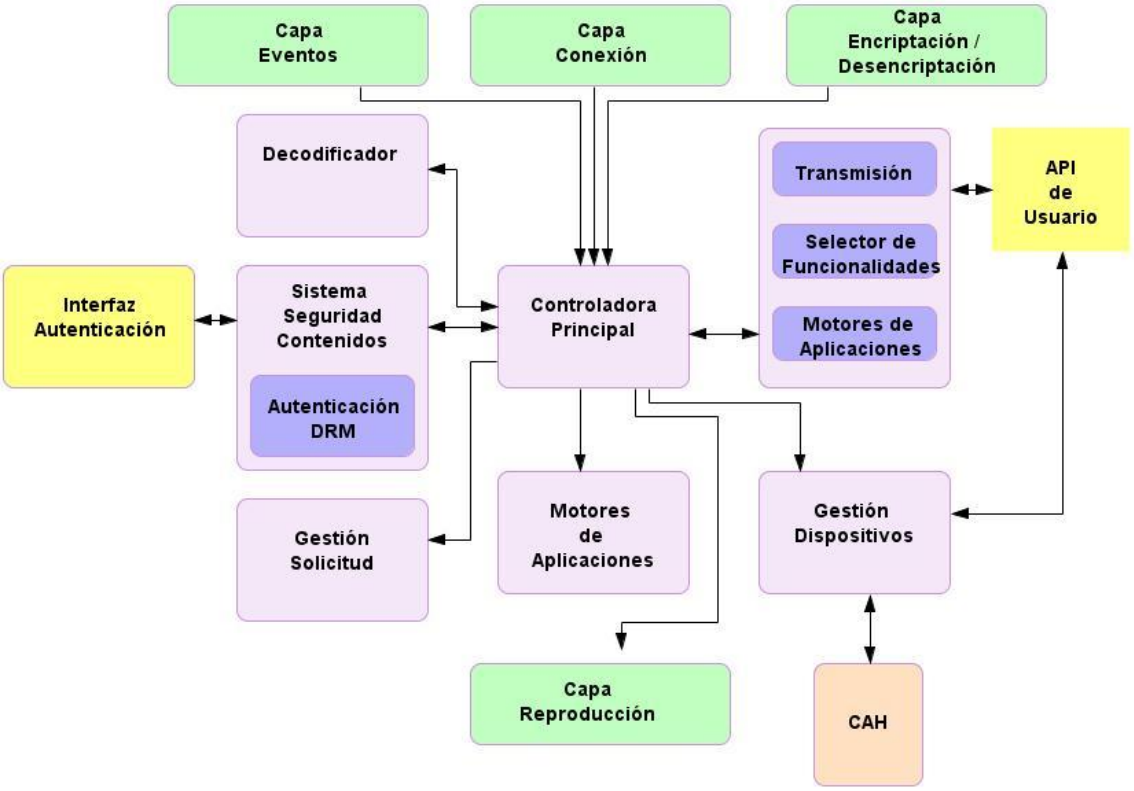


Figura.12 Capa Controladora Principal.

Esta capa implementa las lógicas de negocio relacionadas con la decodificación, transmisión y seguridad del contenido, la gestión de las aplicaciones y dispositivos, la comunicación con la plataforma y las demás capas de la arquitectura, así como la API de Usuario, considerada como la interfaz principal del STB.

- ✓ **Decodificador.**

  Compone la capa controladora principal y se encarga principalmente de la decodificación, demodulación y desmultiplexación del contenido recibido desde la plataforma. Además gestiona el uso de los códec soportados. Esta solución integra MPEG-2, H.264 y AVS.

- ✓ **Sistema de seguridad de Contenidos.**

Este componente incorpora funcionalidades para gestionar la autenticación con el servidor CAS-DRM de la plataforma. Además se encarga de la seguridad de los contenidos y flujos de datos que incorpora el STB.

- ✓ **Componente de Transmisión.**

Encargado de mostrar el contenido disponible al usuario, incorporando un selector de funcionalidades. Además interactúa con el usuario a través de la API de Usuario, interfaz principal del STB y que muestra las funcionalidades disponibles.

La API de Usuario es una incorporación directa de la propuesta de arquitectura de la UIT, adicionando un selector de funcionalidades.

La API de Usuario, según la UIT, ofrece las siguientes funcionalidades:

- ✓ Gestión de contenidos - describe el formato de contenido (por ejemplo, audio, vídeo, subtítulos, etc.) y permite la aplicación de manipular el contenido.
- ✓ Reserva de contenido / grabación - proporciona una interfaz genérica de reserva/grabación. Este API soporta reserva/ grabación de programas de televisión y series.
- ✓ Navegación de contenidos - proporciona un método fácil de hojear el contenido.
- ✓ La reproducción de contenido - según el contenido a reproducir, este API ofrece diferentes controles para contenidos en específico. Por ejemplo, al reproducir un contenido grabado la aplicación tendrá acceso a las API en trick-mode.
- ✓ Exploración / descubrimiento - proporciona interfaces para buscar servicios de multidifusión o emisión de banda ancha, y descubrir redes de contenido de origen similares.

- ✓ Búsqueda de contenidos - proporciona una interfaz para buscar cualquier tipo de contenido, proporcionando filtros específicos para la búsqueda requerida.
- ✓ Plataforma de gestión - API para acceder a la diversas opciones de la plataforma, como la resolución de vídeo, salida de audio, dispositivos de red propia, etc.
- ✓ Gestión de usuarios - la gestión de varios perfiles de usuario, permitiendo a los usuarios personalizar su propio entorno.

Esta interfaz cuenta además con un selector de las funcionalidades disponibles, dígase EPG, televisión, VoD, mensajería instantánea (IM), navegador web, aplicaciones de correo electrónico, aplicaciones VoIP, PVR, video-conferencia, entre otras.

Para ello el componente de transmisión cuenta con una serie de motores de aplicaciones que permiten ejecutar cualquier aplicación independientemente del lenguaje en que hayan sido desarrolladas.

- ✓ **Gestión de dispositivos.**

Agrupa funcionalidades referentes a la configuración e instalación de dispositivos en el STB. Controla los mismos a través de la CAH, e interactúa con el usuario a través de la API de Usuario tratada anteriormente.

- ✓ **Gestión de solicitudes.**

Las solicitudes realizadas a la plataforma IPTV a través de la Capa de Conexión son gestionadas desde este componente. Tanto eventos como peticiones del usuario y de la propia solución son gestionados a través del mismo.

- ✓ **Controlador.**

Es el núcleo de esta capa. Controlador de todos los procesos y componentes antes mencionados y objetivo de atención fundamental a la hora de desarrollar la solución. Implementa lógicas de negocio relacionadas con el sistema a desarrollar.

3.8. Capa de Reproducción.

La capa de reproducción es la encargada de mostrar los contenidos al usuario con la mayor calidad posible contando con la información y configuración necesaria. Además controla los eventos generados por la acción del usuario, gestionados a través de la Capa Controladora de Eventos y enviados a esta para su ejecución en caso de estar relacionados con la configuración de los elementos multimedia.

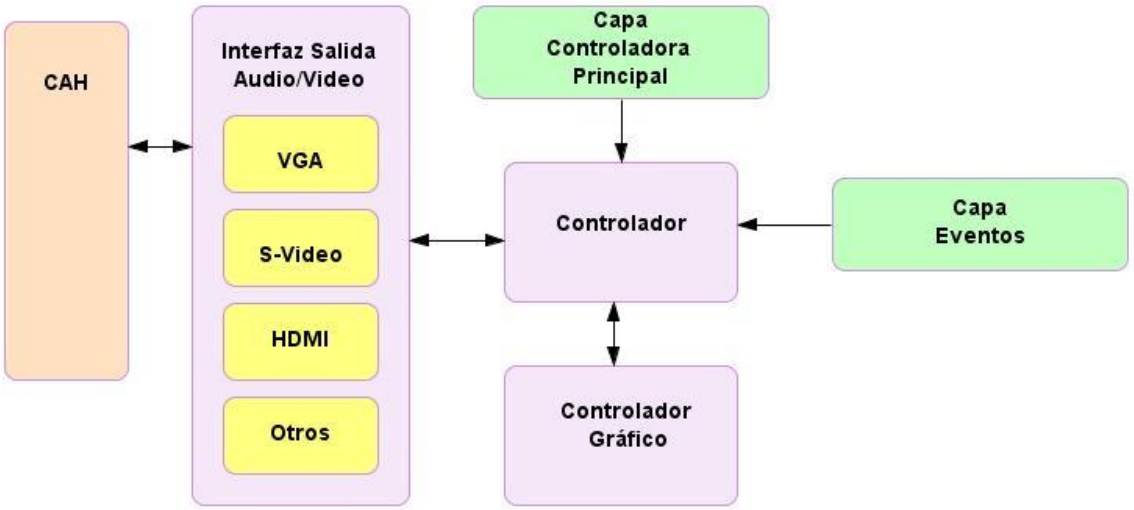


Figura.13 Capa de Reproducción.

Para esto cuenta con un **controlador de gráficos** que se encarga de las configuraciones y opciones correspondientes.

También cuenta con un **controlador** general para gestionar la comunicación con los dispositivos de salida tanto de audio como de video y la acción de eventos del usuario sobre estos. Dicha comunicación es gestionada por las interfaces de VGA, S-Video y HDMI a través de la interacción de estas con la CAH. Además se pueden implementar otras interfaces con el objetivo de ampliar las posibilidades de reproducción del STB.

Esta capa recibe el flujo de contenido a mostrar desde la Capa Controladora Principal y los eventos generados por el usuario y que tienen que ver con la configuración y control de los gráficos del STB generados desde la Capa de Eventos.

3.9. Capa de Eventos.

El objetivo principal de esta capa es recibir las solicitudes y eventos generadas por el usuario del STB y determinar a quien enviarle el mismo para generar la respuesta esperada.

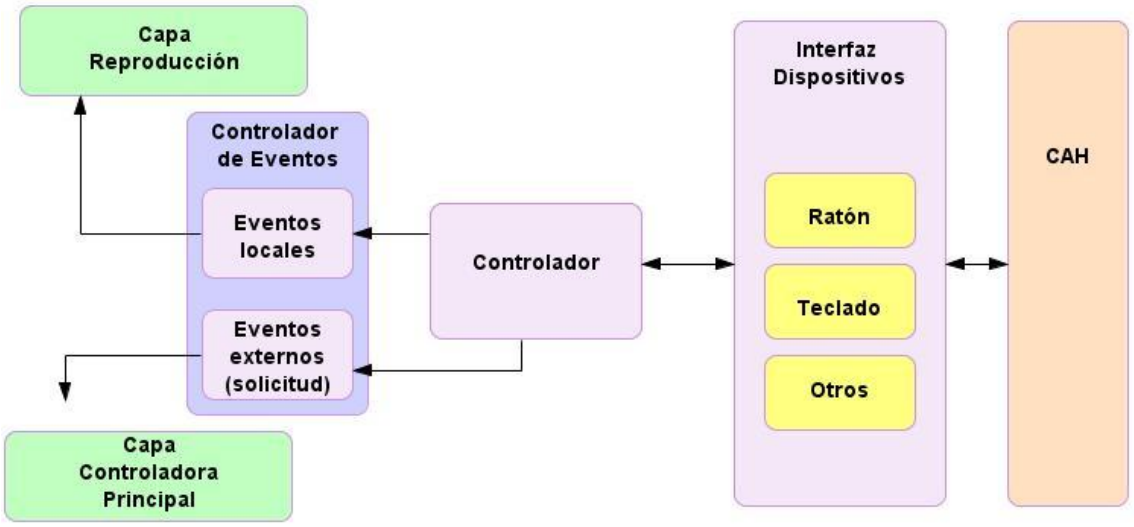


Figura.14 Capa de Eventos.

Su funcionamiento es controlado por un componente encargado de comunicarse con los dispositivos de entrada del ordenador. Para esto cuenta con una interfaz de control para cada dispositivo: mouse, teclado, entre otras que se pueden agregar a la solución según las necesidades.

Además posee un controlador de eventos para gestionar según el tipo de evento a quien enviar la solicitud. Ejemplo de esto es la solicitud por parte del usuario de un contenido o evento cuya respuesta requiere ser gestionada por la plataforma IPTV. Este es enviado a la Capa Controladora Principal para que esta se encargue de realizar la petición correspondiente. No es el caso de los eventos relacionados con gráficos del STB, los cuales se envían a la Capa de Reproducción para ser gestionados localmente.

3.10. Capa de Abstracción de Hardware.

La capa de abstracción de hardware contiene fundamentalmente paquetes de controladores para el hardware específico de cada ordenador. Esta se comunica con todas las capas de la arquitectura propuesta pues representa el intermediario entre estas y el hardware del ordenador. La CAH hace independiente el STB emulado del hardware y esta diseñada para que los controladores de cada dispositivo sean desarrollados si necesidad de preocuparse por la aplicación.

3.11. Evaluación de la Arquitectura.

La calidad de un sistema depende en gran medida de su arquitectura. La evaluación de la misma permite obtener una visión de las decisiones más riesgosas asociadas al software en cuestión y además, aquellas que son buenas y correctas.

3.11.1. ¿Por qué evaluar una arquitectura?

En el campo del software, la arquitectura identifica los elementos más importantes de un sistema así como sus relaciones, es decir da una visión global del sistema. Esto es de suma importancia porque necesitamos la arquitectura para entender el sistema, organizar su desarrollo, plantear la reutilización del software y hacerlo evolucionar.

Por lo tanto cuanto más temprano se encuentre un problema en la arquitectura de un proyecto de software, mejor. El costo de arreglar un error durante las fases de requerimientos o diseño, es mucho menor al costo de arreglar ese mismo error en la fase de verificación. Dado que la arquitectura es un producto temprano de la fase de diseño, esta tiene un profundo efecto en el sistema y en el proyecto [18]. El mal diseño de una arquitectura puede llevar a un proyecto al fracaso. Todos los requerimientos de calidad pueden quedar insatisfechos.

La arquitectura también determina la estructura del proyecto: configuración, presupuesto, alcance, entre otros aspectos. Es mejor cambiar la arquitectura antes que otros artefactos, que están basados en ella, se estabilicen [18].

3.11.2. ¿Cuándo una arquitectura puede ser evaluada?

Generalmente, la evaluación de la arquitectura ocurre después que esta ha sido especificada, pero antes que empiece la implementación. En un proceso iterativo y/o incremental, la evaluación se puede realizar al

final de cada ciclo. Sin embargo, uno de los atractivos de la evaluación de arquitecturas es que se puede efectuar en cualquier etapa de la vida de una arquitectura. En particular, existen dos variaciones útiles: temprana y tardía [18].

3.11.2.1. Evaluación temprana.

La evaluación no tiene por qué esperar a que la arquitectura este totalmente especificada. Esta puede ser utilizada en cualquier etapa del proceso de creación de la arquitectura, para examinar las decisiones arquitectónicas ya tomadas y decidir entre las opciones que están pendientes.

Por supuesto, la completitud y fidelidad de la evaluación es directamente proporcional a la completitud y fidelidad de la descripción de la arquitectura [18].

3.11.2.2. Evaluación tardía.

Esta variación toma lugar no solo cuando la arquitectura está terminada, también cuando la implementación esta completa. Este caso ocurre cuando la organización hereda un sistema legado. La técnica para evaluar un sistema legado es la misma que para evaluar un sistema recién nacido. Una evaluación es útil para entender el sistema legado, y saber si este cumple con los requerimientos de calidad y comportamiento.

En general, una evaluación debe realizarse cuando hay suficiente de la arquitectura como para justificarlo. Una buena regla sería: realizar una evaluación cuando el equipo de desarrollo empieza a tomar decisiones que dependen de la arquitectura y el costo de deshacerlas sobrepasa al costo de realizar una evaluación [18].

3.12. Atributos de la calidad en la Arquitectura.

No es del todo cierto que podamos decir que el sistema alcanzará todas sus metas de calidad con solo mirar la arquitectura. Pero muchos atributos de calidad yacen directamente en el reino de la arquitectura (ver Anexo 1: Atributos de Calidad en la Arquitectura). Una arquitectura puede ser evaluada en base a los siguientes atributos de calidad:

- ✓ **Disponibilidad:** Es la porción de tiempo en que el sistema está levantado y corriendo.
- ✓ **Funcionalidad:** Es la habilidad del sistema de hacer el trabajo para el cual fue construido [19].

- ✓ **Desempeño:** Grado en el cual un sistema o componente cumple con su funcionalidad, dentro de ciertas restricciones dadas, como velocidad, exactitud o uso de memoria [20].
- ✓ **Interoperabilidad:** Es la medida de la habilidad de que un grupo de partes del sistema trabajen con otro sistema.
- ✓ **Portabilidad:** Es la habilidad del sistema de correr sobre diferentes ambientes. Estos ambientes pueden ser de hardware, de software o una combinación de ambos.
- ✓ **Escalabilidad:** Es el grado con el que se pueden ampliar el diseño arquitectónico, de datos o procedimental.
- ✓ **Reusabilidad:** Es la capacidad de diseñar un sistema de forma tal que su estructura o parte de sus componentes puedan ser reutilizados en futuras aplicaciones.

3.13. Técnicas de Evaluación de arquitecturas de software.

Las técnicas existentes en la actualidad para evaluar arquitecturas permiten hacer una evaluación cuantitativa sobre los atributos de calidad a nivel arquitectónico, pero se tienen pocos medios para predecir el máximo (o mínimo) teórico para las arquitecturas de software. Sin embargo, debido al costo de realizar este tipo de evaluación, en muchos casos los arquitectos de software evalúan cualitativamente, para decidir entre las alternativas de diseño [21]. Bosch [21] propone diferentes técnicas de evaluación de arquitecturas de software, a saber: evaluación basada en escenarios, evaluación basada en simulación, evaluación basada en modelos matemáticos y evaluación basada en experiencia.

3.13.1. Evaluación basada en escenarios.

De acuerdo con Kazman [22] un escenario es una breve descripción de la interacción de alguno de los involucrados en el desarrollo del sistema con éste. Este consta de tres partes: el estímulo, el contexto y la respuesta. El estímulo es la parte del escenario que explica o describe lo que el involucrado en el desarrollo hace para iniciar la interacción con el sistema. Puede incluir ejecución de tareas, cambios en el sistema, ejecución de pruebas, configuración, etc. El contexto describe qué sucede en el sistema al momento del estímulo. La respuesta describe, a través de la arquitectura, cómo debería responder el sistema ante el estímulo. Este último elemento es el que permite establecer cuál es el atributo de calidad asociado.

Los escenarios proveen un vehículo que permite concretar y entender atributos de calidad. Entre las ventajas de su uso están:

- ✓ Son simples de crear y entender.
- ✓ Son poco costosos y no requieren mucho entrenamiento.
- ✓ Son efectivos.

Actualmente las técnicas basadas en escenarios cuentan con dos instrumentos de evaluación relevantes, a saber: el *Utility Tree* propuesto por Kazman [22], y los *Profiles,* propuestos por Bosch [21].

3.13.2. Evaluación basada en simulación.

Bosch [21] establece que la evaluación basada en simulación utiliza una implementación de alto nivel de la arquitectura de software.

El proceso de evaluación basada en simulación sigue los siguientes pasos [21]:

- ✓ Definición e implementación del contexto.
- ✓ Implementación de los componentes arquitectónicos.
- ✓ Implementación del perfil.
- ✓ Simulación del sistema e inicio del perfil.
- ✓ Predicción de atributos de calidad.

La exactitud de los resultados de la evaluación depende, a su vez, de la exactitud del perfil utilizado para evaluar el atributo de calidad y de la precisión con la que el contexto del sistema simula las condiciones del mundo real.

3.13.3. Evaluación basada en modelos matemáticos.

Bosch establece que la evaluación basada en modelos matemáticos se utiliza para evaluar atributos de calidad operacionales. Permite una evaluación estática de los modelos de diseño arquitectónico, y se presentan como alternativa a la simulación, dado que evalúan el mismo tipo de atributos. Ambos enfoques pueden ser combinados, utilizando los resultados de uno como entrada para el otro [21].

Los siguientes pasos son los que sigue esta técnica para la evaluación:

- ✓ Selección y adaptación del modelo matemático.
- ✓ Representación de la arquitectura en términos del modelo.
- ✓ Estimación de los datos de entrada requeridos.
- ✓ Predicción de atributos de calidad.

3.13.4. Evaluación basada en experiencia.

Bosch establece que en muchas ocasiones los arquitectos e ingenieros de software otorgan valiosas ideas que resultan de utilidad para la evasión de decisiones erradas de diseño. Aunque todas estas experiencias se basan en evidencia anecdótica; es decir, basada en factores subjetivos como la intuición y la experiencia. Sin embargo, la mayoría de ellas puede ser justificada por una línea lógica de razonamiento, y puede ser la base de otros enfoques de evaluación [21]. Existen dos tipos de evaluación basada en experiencia: la evaluación informal, que es realizada por los arquitectos de software durante el proceso de diseño, y la realizada por equipos externos de evaluación de arquitecturas.

3.14. Métodos de Evaluación de la Arquitectura.

3.14.1. SAAM – Software Architecture Analysis Method.

SAAM es un método para la evaluación de Arquitecturas basado en escenarios que se centra para ello, principalmente, en el atributo de calidad modificabilidad. Este método ha demostrado en la práctica que es útil para evaluar otros atributos de calidad de forma rápida tales como portabilidad, extensibilidad, integrabilidad, así como el cubrimiento funcional que tiene la arquitectura sobre los requerimientos del sistema.

SAAM puede ser utilizado para evaluar una o más arquitecturas. Si son evaluadas varias arquitecturas el mismo permite la comparación entre estas obteniendo como resultado final una tabla con las fortalezas y debilidades de cada una en los distintos escenarios. Si se evalúa una sola se obtendrá como resultado final un reporte con los componentes computacionales donde la arquitectura no alcanza el nivel requerido. En ninguno de los casos anteriores se emite un valor definitivo acerca de la calidad arquitectónica.

Roles de SAAM:

Los roles que se identificaron para este método son los siguientes:

- ✓ Interesados externos (Organización cliente, usuarios finales, administradores del sistema, etc.)
- ✓ Interesados internos (Arquitectos de Software, analistas de requerimientos)
- ✓ Equipo SAAM (Líder, expertos en el dominio de la aplicación, expertos externos en arquitectura)

Metodología:

- ✓ **Paso 1. Desarrollo de escenarios.**

  La meta principal es capturar las principales actividades que el sistema debe soportar. Los escenarios encontrados deben reflejar los atributos de calidad de interés y también mostrar la interacción entre las diferentes personas que utilizarán el sistema: usuarios finales, administrador, mantenedor, desarrolladores, etc.

- ✓ **Paso 2. Descripción de la Arquitectura.**

  En este paso se presentan las arquitecturas candidatas. . Se deben indicar los principales elementos de la arquitectura.

- ✓ **Paso 3. Clasificación de escenarios.**

  Se clasifican los escenarios en directos o indirectos (es directo si la arquitectura puede soportar el escenario sin cambios).

- ✓ **Paso 4. Evaluación de escenarios.**

  Para cada escenario indirecto, se deben listar los cambios necesarios en la arquitectura para soportarlo, y el costo de llevarlos a cabo debe ser estimado.

- ✓ **Paso 5. Interacción de escenarios.**

  Se deben determinar las interacciones de escenarios sobre cada componente de cada arquitectura.

- ✓ **Paso 6. Evaluación general.**

Un peso es asignado a cada escenario en términos de su influencia para que el sistema sea exitoso. El peso puede ser elegido de acuerdo a los objetivos del negocio, costos, riesgos, etc.

Fortalezas y Debilidades:

Las principales fortalezas que posee este método son:

- ✓ Los interesados comprenden con facilidad las arquitecturas evaluadas.
- ✓ La documentación es mejorada.
- ✓ El esfuerzo y el costo de los cambios pueden ser estimados con anticipación.
- ✓ Analogía con el concepto de bajo acoplamiento y alta cohesión.

Las debilidades son:

- ✓ La generación de escenarios está basada en la visión de los interesados.
- ✓ No provee una métrica clara sobre la calidad de la arquitectura Evaluada.

3.14.2. ATAM – Architecture Tradeoff Analysis Method.

ATAM es un método que especifica cuán bien satisface una arquitectura los atributos de calidad, además de que provee ideas de cómo interactúan y realizan concesiones mutuas esos atributos de calidad entre sí. Este método puede ser utilizado además para analizar sistemas legados cuando estos necesitan ser integrados con otros sistemas, modificados, etc.

Pasos del ATAM:

El método consta de nueve pasos, divididos en cuatro grupos:

- ✓ **Presentación:** donde la información es intercambiada.
- ✓ **Investigación y Análisis:** donde se valoran los atributos claves de calidad requeridos, uno a uno con las propuestas arquitectónicas.

- ✓ **Pruebas:** donde se revisan los resultados obtenidos contra las necesidades relevantes de los clientes.
- ✓ **Informes:** donde se presentan los resultados de ATAM.

Presentación:

Paso 1: Presentar el ATAM [18].

- ✓ Los pasos del ATAM en resumen.
- ✓ Las técnicas que serán utilizadas para la obtención y análisis.
- ✓ Las salidas de la evaluación.

Paso 2: Presentar las pautas del negocio.

- ✓ Las funciones más importantes del sistema.
- ✓ Toda restricción técnica.
- ✓ Las guías de la arquitectura.

Paso 3: Presentar la arquitectura.

- ✓ Las restricciones técnicas.
- ✓ Otros sistemas.
- ✓ Propuestas Arquitectónicas.

Investigación y Análisis

Paso 4: Identificar las propuestas arquitectónicas.

- ✓ Las propuestas arquitectónicas son identificadas por el arquitecto, pero no son analizadas.

Paso 5: Generar el árbol de utilidad de los atributos de calidad.

- ✓ Los atributos de calidad que comprometen la utilidad del sistema (*performance*, *availability*, entre otros) son obtenidos, especificados en escenarios (con estímulos y respuestas) y priorizados. (ver Anexo 2 Árbol de utilidad)

Paso 6: Analizar las propuestas arquitectónicas.

- ✓ Basados en los escenarios de mayor prioridad identificados en el paso 5, las propuestas arquitectónicas que cumplen con estos escenarios, son obtenidas y analizadas (por ejemplo, una propuesta arquitectónica que logra una meta de performance, será objeto de un análisis de *performance*).
- ✓ Durante este paso los riesgos arquitectónicos, los no riesgos, los *sensitivity points* y *tradeoff points* son identificados.

Paso 7: Lluvia de ideas y priorización de los escenarios.

Este paso consiste en la generación de nuevos escenarios para:

- ✓ Representar los intereses de los clientes que no hayan sido comprendidos;

Paso 8: Analizar las propuestas arquitectónicas.

- ✓ Se reitera las actividades del paso 6 utilizando el ranking de escenarios del paso 7.
- ✓ Estos escenarios se consideran casos de prueba para confirmar el análisis realizado hasta ahora.

Paso 9: Presentar los resultados.

- ✓ El documento de propuestas arquitectónicas.
- ✓ El conjunto de escenarios priorizados.
- ✓ El árbol de utilidad.
- ✓ Los riesgos descubiertos.
- ✓ Los no riesgos documentados.

- ✓ Los *sensitivity points* y *tradeoff points* encontrados.

3.14.3. ARID-An Evaluation Method for Partial Architecture.

ARID es utilizado cuando es conveniente realizar la evaluación de diseños parciales en las etapas tempranas del desarrollo. Hacer revisiones en las etapas intermedias provee una valiosa visión de la viabilidad de la arquitectura a construir, además de que permite descubrir errores e inconsistencias en la misma. El método ARID cumple con estas características y se basa en las mejores cualidades de los métodos basados en escenarios (ATAM o SAAM) y las revisiones activas de diseños (ADRS).

De los ADRS toma la participación activa de los entrevistados lo cual es ideal por dos motivos:

- ✓ Asegura alta fidelidad en las respuestas.
- ✓ La participación activa puede captar la atención del grupo de compradores.

Del ATAM se queda con la idea de la generación de los escenarios por parte de todos los interesados, los usuarios le dirán a los diseñadores que es lo que ellos necesitan o mejor dicho que es lo que ellos esperan y si los diseñadores demuestran en la pruebas que el diseño cumple con los requerimientos también va a atraer el interés de los compradores.

¿Qué es ADRS? [18]

ADRS son técnicas efectivas para asegurar la calidad dando diseños detallados del software.

ADR es utilizado para la evaluación de diseños detallados de unidades del software como los componentes o módulos. Las preguntas giran en torno a la calidad y completitud de la documentación y la suficiencia, el ajuste y la conveniencia de los servicios que provee el diseño propuesto.

Roles en ARID:

- ✓ Equipo de Verificación (líder de evaluación, arquitecto).
- ✓ Arquitecto.
- ✓ Revisores.

Pasos de ARID:

ARID está compuesto por 9 pasos separados en dos fases.

- ✓ **Fase 1:** Pre reunión: Reunión entre el arquitecto y el líder de la evaluación.
- ✓ **Fase 2:** Evaluación: Los revisores son reunidos y la evaluación comienza.

FASE 1:

- ✓ **Identificar los revisores:** ingenieros de software que van a usar el diseño.
- ✓ **Preparar la presentación del diseño:** el arquitecto prepara un informe que explica el diseño, el mismo deberá ser lo suficientemente detallado como para que una capacitada audiencia pueda usar el diseño.
- ✓ **Preparar los escenarios:** el arquitecto y el líder preparan los escenarios bases que sirven para ilustrar los conceptos a los revisores.
- ✓ **Preparar los materiales:** copias de la presentaciones, escenarios.

FASE 2:

- ✓ **Presentación del método:** el líder utiliza 30 minutos para explicar los pasos de la evaluación a los participantes.
- ✓ **Presentación del diseño:** el arquitecto realiza una presentación del diseño mostrando los ejemplos.
- ✓ **Lluvia de ideas y priorización de escenarios:** se proponen escenarios relevantes para solucionar problemas.
- ✓ **Aplicación de los escenarios:** El líder del grupo de revisión es el encargado de probar que el diseño provea los escenarios, comenzando por los de mayor prioridad, hasta que concluya el tiempo estimado de la evaluación, se hayan analizado los escenarios de mayor prioridad, o la conclusión de la revisión satisfaga a los implicados.

- ✓ **Resumen:** Al final, el facilitador recuenta la lista de puntos tratados, pide opiniones de los participantes sobre la eficiencia del ejercicio de revisión, y agradece por su participación.

Estos pasos se pueden ver más argumentados en el Anexo 3: Pasos del ARID.

3.15. Evaluando la Arquitectura

Si las decisiones arquitectónicas determinan los atributos de calidad del sistema, entonces es posible evaluar dichas decisiones con respecto a su impacto sobre estos atributos.

Teniendo en cuenta lo anterior expuesto y que el Proyecto IPTV de la Facultad 2 se encuentra en las primeras faces del proceso de desarrollo de software se utilizará ARID, ya que permite realizar evaluación de la arquitectura de diseños parciales en etapas tempranas del desarrollo. Es sencillo de utilizar y el costo de evaluación es realmente bajo.

Los arquitectos del sistema tienen un dominio completo del diseño arquitectónico del mismo, y presentan conocimientos del método a aplicar para la evaluación, por lo que no se realizó la Fase1, pasando directamente a aplicarse los tres últimos pasos de la Fase 2.

A continuación se exponen los escenarios seleccionados por su interés dentro de la arquitectura:

Escenario # 1	Seguridad de los datos en la Plataforma IPTV.
Atributo	Integridad, Confiabilidad.
Estímulo	El cliente solicita acceder a un contenido.
Respuesta	El STB realiza la petición correspondiente a la Plataforma y muestra el contenido solicitado al usuario.
Contexto	
Al ejecutar el emulador de STB este establece conexión con la plataforma, posteriormente descarga la información de los contenidos disponibles y los muestra al usuario a través de una interfaz. Luego el usuario selecciona el contenido que desea percibir lo que genera una nueva petición a la plataforma. Esta petición se realiza a través del protocolo HTTPS, en caso de ser atendida se establece una sesión IGMP o RTSP, mediante los cuales se accede a los contenidos de difusión y VoD respectivamente. En	

estos casos así como en las peticiones HTTPS el contenido viaja encriptado a través de la red y sólo podrá ser percibido por el usuario si posee los permisos necesarios para desencriptar el mismo. Lo anterior expuesto demuestra la plena seguridad, integridad y confiabilidad de los datos en cualquier flujo de transmisión entre el STB y la plataforma IPTV.

Tabla.1 Escenario # 1

Escenario # 2	Ejecutar la solución en una PC que carece de los requerimientos mínimos de Hardware.
Atributo	Funcionalidad, Eficiencia.
Estímulo	Instalar o iniciar la aplicación.
Respuesta	La aplicación muestra un mensaje informando al usuario que la PC no cumple con los requerimientos mínimos de Hardware necesario dando la opción de continuar el proceso o detener el mismo a consideración del usuario.
Contexto	
Al instalar o ejecutar el emulador de STB en una PC, este verifica que la misma cuente con las condiciones de hardware necesarias para ejecutar dicha acción. Los requerimientos mínimos de hardware están reflejados en los requerimientos no funcionales de esta propuesta. En caso que no se cumplieran los mismos el usuario tiene la opción de continuar el proceso, hecho que refleja la alta funcionalidad y eficiencia de la solución.	

Tabla.2 Escenario # 2

Escenario # 3	Aumento de dispositivos de salida
Atributo	Modificabilidad, Escalabilidad, Flexibilidad.
Estímulo	Agregar dispositivo de salida (AV)
Respuesta	Implementación de una nueva interfaz en la Capa de Reproducción.

Contexto
El estilo de arquitectura en capa proporciona una organización jerárquica de manera que cada capa interactúa con la capa superior e inferior. Esta están formadas por paquetes o subsistemas que implementan funcionalidades afines entre si. Agregar un nuevo dispositivo de salida al STB, implica modificar el componente que se encarga del control de los dispositivos para lo que es necesario la implementación de una nueva interfaz que provea los medios para comunicarse con dicho dispositivo. Como se puede observar los componentes afectados son mínimos y relacionados con la configuración del sistema, por lo que resalta la configurabilidad, flexibilidad, modificabilidad y por tanto el mantenimiento del sistema.

Tabla.3 Escenario # 3

Escenario # 4	Incremento de la capacidad de codificación.
Atributo	Reusabilidad, Modificabilidad.
Estímulo	Agregar Códec de video al emulador.
Respuesta	Sistema integra el nuevo códec al componente de codificación.
Contexto	
El decodificador es uno de los componentes que conforma la capa controladora principal. Dentro de sus funcionalidades se encuentra la demodulación, desmultiplexación y decodificación de los contenidos. En un momento dado se pueden instalar nuevos códec sin necesidad de cambiar la lógica del negocio de dicho componente. Sin embargo se amplía la funcionalidad del emulador de STB considerablemente.	

Tabla.4 Escenario # 4

Al ser llevado a cabo el paso 3 de la Fase 2 se observaron algunos escenarios del sistema y su relación con los atributos de calidad. Algunos no se pudieron evaluar, porque es imposible en este estado, determinar si el sistema contará con determinadas características, a la hora de brindar una funcionalidad prevista. Sin embargo los que fueron analizados son aquellos que recibieron mayor

votación por los involucrados en el proceso de evaluación, presentándose en el documento los más relevantes dentro de estos.

A partir de todos los elementos que se han reflejado, se puede afirmar que la solución cumple con los atributos Integridad, Confiabilidad, Funcionalidad, Eficiencia, Modificabilidad, Escalabilidad, Flexibilidad, Reusabilidad.

Teniendo en cuenta lo expuesto anteriormente, se define de manera general y a partir de las consideraciones generadas, como resultado de la combinación de los atributos contemplados que el diseño arquitectónico evaluado de manera parcial, dentro de una etapa temprana del desarrollo, resulta adecuado.

3.16. Conclusiones

En este capítulo se realizó la propuesta de arquitectura basada en el estilo de Arquitectura en Capas, definiendo las principales funcionalidades y requisitos no funcionales de la solución. Se definieron y describieron las capas que conforman la propuesta relacionando componentes e interfaces. Además se llevó cabo la evaluación de la arquitectura propuesta, llegando a la conclusión de que la misma satisface los atributos de calidad evaluados. Por lo anteriormente descrito se considera que esta arquitectura puede ser usada para el Emulador de Set Top Box para plataforma IPTV.

CONCLUSIONES

Teniendo en cuenta la importancia que tienen las arquitecturas en el proceso de desarrollo de software se propuso un diseño arquitectónico para un Emulador de Set Top Box para la plataforma IPTV a desarrollar en la UCI, que cumple con los objetivos trazados para esta investigación.

La arquitectura que se propone cumple con los requerimientos y contribuciones estandarizados por la UIT. Además satisface gran parte de las problemáticas detectadas al realizar el estado del arte.

Con vista a lograr un diseño arquitectónico que permita la partición del problema en cuestión en una secuencia de pasos incrementales se decidió utilizar una arquitectura por capas, la cual proporciona amplia reutilización de los componentes y subsistemas propuestos.

Para la validación de la propuesta se utilizó el método de evaluación de arquitecturas ARID, el cual permitió evaluar determinados atributos de calidad concluyendo que el diseño arquitectónico propuesto cumple con los requerimientos no funcionales definidos, cubriendo las necesidades y funcionalidades del Emulador de STB en la plataforma IPTV a desarrollar en la UCI.

RECOMENDACIONES

Las metas planteadas en este trabajo se han cumplido en base a los resultados obtenidos y de un conjunto de ideas surgidas a lo largo de la investigación que no están enmarcadas dentro de los objetivos de este trabajo, aunque sí forman parte del área de investigación, se recomienda:

- ✓ Mantener esta propuesta en constante refinamiento durante el ciclo de desarrollo.
- ✓ Realizar una revisión mas detallada de cada uno de los escenarios de la aplicación para evaluar la arquitectura propuesta.
- ✓ Realizar un estudio de factibilidad y una evaluación de los costos de la aplicación a desarrollar.

BIBLIOGRAFÍA

1. **O'Driscoll, Gerard**. *Next Generation IPTV Services and Technologies.* WILEY, 2007.
2. **Guía Práctica para el usuario,** Centro Virtual de capacitación a distancia de la UIT. Introducción a IPTV s.n., 2004-2005.
3. **Ramirez, David.** *IPTV security. Protecting High Value Digital Contents.* WILEY, 2008.
4. **Guía Práctica para el usuario,** Centro Virtual de capacitación a distancia de la UIT. Middleware. s.n., 2004-2005.
5. **IEEE. 2004.** IEEE Standards Association. *IEEE STD 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.* [Online] 2004. [Cited: 12 2, 2009.] http://standards.ieee.org/reading/ieee/std_public/description/se/1471-2000_desc.html.
6. **Jacobson, Ivar, Booch, Grady and Rumbaugh, James. 2000.** *El proceso unificado de desarrollo de software.* Madrid: Addison Wesley, 2000.
7. **Hofmeister, C, Nord, R and Soni, D. 2000.** *Applied Software Architecture.* s.l.: Addison, 2000.
8. **IEEE. 2004.** IEEE Standards Association. *IEEE Std 1471-2000 IEEE Recommended Practice for Architectural Description of Software-Intensive Systems.* [Online] 2004. [Cited: 12 2, 2009.] http://standards.ieee.org/reading/ieee/std_public/description/se/1471-2000_desc.html.
9. **Pressman, Roger S.** Capítulo 14. Diseño arquitectónico. *Ingeniería del Software. Un enfoque práctico.* Madrid: Mc Graw Hill, 2001, pág. 237-258.
10. **Reynoso, Carlos y Kicillof, Nicolás.** *Estilos y Patrones en la Estrategia de Arquitectura de Microsoft.* UNIVERSIDAD DE BUENOS AIRES: s.n., 2004. Versión 1.0.
11. **Edvdemon, John.** Principios de diseños de servicios: patrones y antipatrones de servicios. *www.microsoft.com* [En línea] Agosto de 2006. [Citado el: 22 de Enero de 2010.] http://www.microsoft.com/spanish/msdn/articulos/archivo/121205/voices/SOADesign.mspx.
12. **Scheneider, Jean Guy.** *Components, Scripts and Glue: A conceptual Framework for Software Composition. Tesis Doctoral.* Intitut fur Informatik und angewandte Mathematik, Universitat Bem: s.n., 1999.
13. **Shaw, Mary y Garlan, David.** *Architecture Software, Perspectives on an emerging discipline.* s.l.: Prentice Hall. 1996.

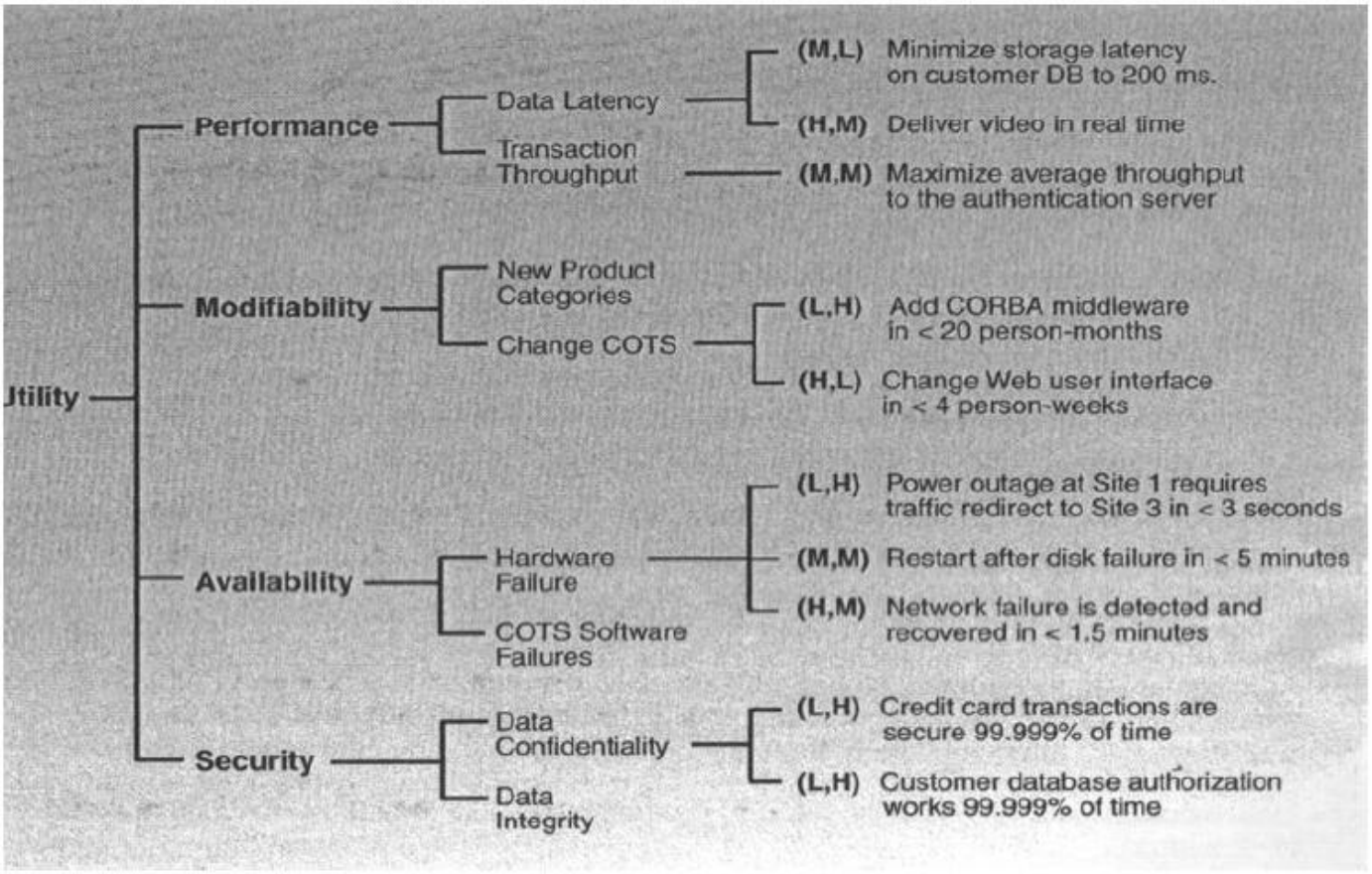
14. *Lenguajes de descripción arquitectónica de software (ADL).* **Reinoso, Carlos y Kicillof, Nicolás.** Universidad de Buenos Aires: s.n., Marzo 2004.
15. **Allen y J., Robert.** *A Formal Approach to Software Architecture.* School of Computer Science Carnegie Mellon University, Pittsburgh: s.n., 1997. CMU-CS-97-144.
16. **Buschmann, F, et al. 1996.** *Pattern Oriented Software Architecture. A system of patterns.* Inglaterra: John Wiley & Sons, 1996.
17. **Kaiser, S H. 2005.** *Software Paradigms.* 2005.
18. **Dávila Mauricio, Martín Germán, Diego Crutas, Andrés García.** Evaluación de arquitecturas de software. *Facultad de Ingeniería Gestión de Software.*
19. **Kazman, R., Clements, P., Klein, M.** *Evaluating Software Architectures. Methods and case studies.* 2001.
20. **IEEE.** IEEE standard glossary of software engineering terminology. *IEEE Std 610.12.* 12 de Diciembre de 1990.
21. **Bosch, J.** *Design & Use of Software Architectures.* 2000.
22. **Kazman, R., Clements, P., Klein, M.** *Evaluating Software Architectures.* 2001.
23. **IEEE-1471.** *Recommended Practice for. Architecture lDescription of Software-Intensive Systems.* 2000.
24. **IBM Corp.** *Ayuda del Rational.* s.l.: IBM *Corporation* 1987, 2006.
25. **FG IPTV-C-0583.** *Proposal fora STB IPTV middleware generic architecture and User API.* Bled, Eslovenia. May, 7-11, 2007.

ANEXOS

Anexo 1: Atributos de Calidad en la Arquitectura.

IEEE Std 1061	ISO Std 9126	MITRE Guide to Total Software Quality Control	
Eficiencia	Funcionalidad	Eficiencia	Integridad
Funcionalidad	Confiabilidad	Confiabilidad	Supervivenciabilidad
Mantenibilidad	Usabilidad	Usabilidad	Corretitud
Portabilidad	Eficiencia	Mantenibilidad	Verificabilidad
Confiabilidad	Mantenibilidad	Expandibilidad	Flexibilidad
Usabilidad	Portabilidad	Interoperabilidad	Portabilidad
		Reusabilidad	

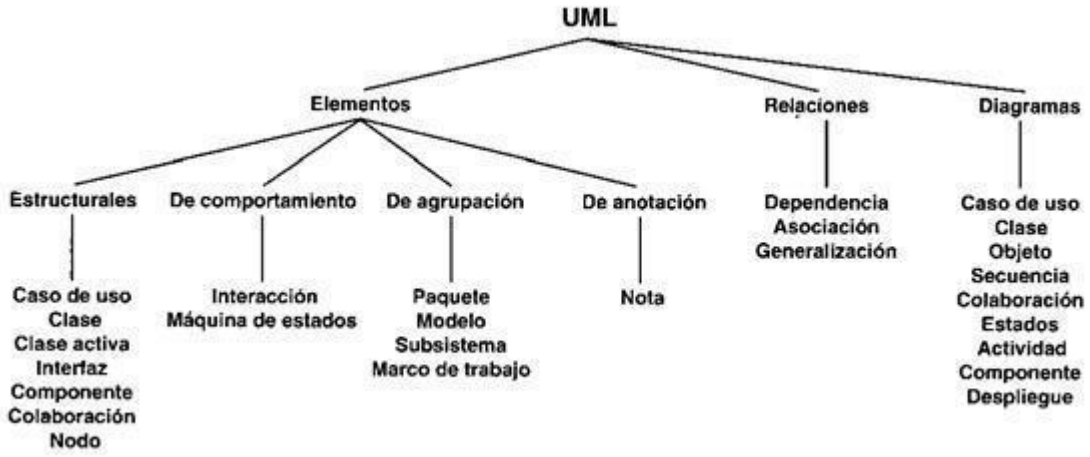
Anexo 2: Árbol de Utilidad.



Anexo 3: Pasos de ARID.

Fase 1: Actividades Previas	
1. Identificación de los encargados de la revisión	Los encargados de la revisión son los ingenieros de software que se espera que usen el diseño, y todos los involucrados en el diseño. En este punto, converge el concepto de *encargado de revisión* de ADR e *involucrado* del ATAM.
2. Preparar el informe De diseño	El diseñador prepara un informe que explica el diseño. Se incluyen ejemplos del uso del mismo para la resolución de problemas reales. Esto permite al facilitador anticipar el tipo de preguntas posibles, así como identificar áreas en las que la presentación puede ser mejorada.
3. Preparar los escenarios base	El diseñador y el facilitador preparan un conjunto de escenarios base. De forma similar a los escenarios del ATAM y el SAAM, se diseñan para ilustrar el concepto de escenario, que pueden o no ser utilizados para efectos de la evaluación.
4. Preparar los materiales	Se reproducen los materiales preparados para ser presentados en la segunda fase. Se establece la reunión, y los involucrados son invitados.
Fase 2: Revisión	
5. Presentación del ARID	Se explica los pasos del ARID a los participantes.
6. Presentación del diseño	El líder del equipo de diseño realiza una presentación, con ejemplos incluidos. Se propone evitar preguntas que conciernen a la implementación o argumentación, así como alternativas de diseño. El objetivo es verificar que el diseño es conveniente.
7. Lluvia de ideas y establecimiento de prioridad de escenarios	Se establece una sesión para la lluvia de ideas sobre los escenarios y el establecimiento de prioridad de escenarios. Los involucrados proponen escenarios a ser usados en el diseño para resolver problemas que esperan encontrar. Luego, los escenarios son sometidos a votación, y se utilizan los que resultan ganadores para hacer pruebas sobre el diseño.
8. Aplicación de los escenarios	Comenzando con el escenario que contó con más votos, el facilitador solicita pseudo-código que utiliza el diseño para proveer el servicio, y el diseñador no debe ayudar en esta tarea. Este paso continúa hasta que ocurra alguno de los siguientes eventos: ✓ Se agota el tiempo destinado a la revisión ✓ Se han estudiado los escenarios de más alta prioridad ✓ El grupo se siente satisfecho con la conclusión alcanzada. Puede suceder que el diseño presentado sea conveniente, con la exitosa aplicación de los escenarios, o por el contrario, no conveniente, cuando el grupo encuentra problemas o deficiencias.
9. Resumen	Al final, el facilitador recuenta la lista de puntos tratados, pide opiniones de los participantes sobre la eficiencia del ejercicio de revisión, y agradece por su participación.

Anexo 4: Componentes UML.



GLOSARIO DE TÉRMINOS

ADSL: *Asymmetric Digital Subscriber Line.* (Línea de Suscripción Digital Asimétrica).

API: *Application Programming Interface*. (Interfaz de Programación de Aplicaciones).

ATSC: *Advanced Television System Committee*. (Grupo encargado del desarrollo de los estándares de la televisión digital en los Estados Unidos).

BLUE RAY: Formato de disco óptico de nueva generación de 12 cm de diámetro para videos de alta definición.

CATV: Televisión por cable.

DHCP: *Dynamic Host Configuration Protocol*. (Protocolo de Configuración Dinámica de Host).

DSSA: Arquitectura de Software de Dominio Específico.

DSLAM: *Digital Subscriber Line Access Multiplexer.* (Multiplexor de acceso a la línea digital de abonado).

DRM: *Digital Rights Management*. (Gestión de Derechos Digitales).

DTMB: *Digital Terrestrial Multimedia Broadcast.(*Estándar de televisión digital terrestre para terminales fijos y móviles de la República Popular China. Adoptado como estándar nacional el 2006).

DVB-SI: *Digital Video Broadcasting-Service Information*. (Estándar de transmisión de datos en las emisiones de televisión digital).

DVB-S: Sistema que permite incrementar la capacidad de transmisión de datos y televisión digital a través de un satélite usando el formato MPEG2.

DVD: *Digital Versatile Disc*. (Disco versátil Digital. El DVD es un dispositivo de almacenamiento óptico estandarizado).

ECMAScript: Especificación de lenguaje de programación publicada por ECMA International.

EPG: Guía de Programación Electrónica.

ETECSA: Empresa de Telecomunicaciones de Cuba SA.

MIDDLEWARE: Software de conectividad que ofrece un conjunto de servicios que hacen posible el funcionamiento de aplicaciones distribuidas sobre plataformas heterogéneas.

HDMI: *High-Definition Multimedia Interface*. (Interfaz multimedia de alta definición).

HTML: *HyperText Markup Language.* (Lenguaje de Marcado de Hipertexto).

HTTPS: *Hypertext Transfer Protocol Secure*.

H.264: H.264 o MPEG-4 parte 10 es una norma que define un códec de vídeo de alta compresión, desarrollada conjuntamente por el ITU-T *Video Coding Experts Group* (VCEG) y el ISO/IEC *Moving Picture Experts Group* (MPEG).

IGMP: *Internet Group Management Protocol*. (El protocolo de red IGMP se utiliza para intercambiar información acerca del estado de pertenencia entre enrutadores IP que admiten la multidifusión y miembros de grupos de multidifusión).

IP: Internet Protocol.

IPTV: *Internet Protocol Television*.

ISDB: Conjunto de normas creado por Japón para las transmisiones de radio digital y televisión digital.

LAN: *Local Area Network,* red de área local.

MPEG-2/4: Moving Pictures Experts Group 2/4 (MPEG-2/4).

NAT: *Network Address Translation.* (Traducción de Dirección de Red).

NTSC: *National Television System Committee*. (Comisión Nacional de Sistemas de Televisión. Es un sistema de codificación y transmisión de Televisión en color analógico).

PAL: *Phase Alternating Line.* (Línea de fase alternada. Es el nombre con el que se designa al sistema de codificación utilizado en la transmisión de señales de televisión analógica en color en la mayor parte del mundo).

PC: Computadora Personal.

PS3: PlayStation 3.

PVC: P*ersonal Video Coder*. (Cámara de Grabación de Video).

QoS: Calidad de Servicio.

RTSP: *Real Time Streaming Protocol*. (El protocolo de flujo de datos en tiempo real establece y controla uno o muchos flujos sincronizados de datos, ya sean de audio o de video).

RUP: *Rational Unified Process*. (Proceso Unificado de Racional).

SECAM: Es un sistema para la codificación de televisión en color analógica utilizado por primera vez en Francia.

SOA: Arquitectura Orientada a Servicios.

STB: Set Top Box, es el nombre con el que se conoce el dispositivo encargado de la recepción y opcionalmente decodificación de señal de televisión analógica o digital para luego ser mostrada en un dispositivo de televisión.

SVG: *Scalable Vector Graphics.* (Vector Gráfico Escalable).

S-Video: *Separate-Video*. (Video separado).

TCP/IP: Conjunto de protocolos de red en los que se basa Internet y que permiten la transmisión de datos entre redes de computadoras.

TIC: Tecnologías de la información y la comunicaciones.

TV: Televisión. Es un sistema para la transmisión y recepción de imágenes en movimiento y sonido a distancia. El receptor de las señales es el televisor.

TVD: Televisión digital. Se refiere al conjunto de tecnologías de transmisión y recepción de imagen y sonido, a través de señales digitales.

VGA: *Video Graphics Array.* (VGA se refiere a una pantalla analógica estándar de ordenadores, cuyo conector también representa un estándar).

VLAN: Virtual LAN. (Red de área local virtual).

VoD: Video bajo demanda.

UCI: Universidad de las Ciencias Informáticas.

UIT: Unión Internacional de Telecomunicaciones.

UML: Lenguaje Unificado de Modelado.