ESCUELA POLITÉCNICA NACIONAL

FACULTAD DE INGENIERÍA ELÉCTRICA Y ELECTRÓNICA

DISEÑO E IMPLEMENTACIÓN DE APLICACIONES INTERACTIVAS BASADAS EN GINGA- NCL PARA TELEVISIÓN DIGITAL EN EL ÁREA DE EDUCACIÓN SUPERIOR

PROYECTO PREVIO A LA OBTENCIÓN DEL TÍTULO DE INGENIERO EN ELECTRÓNICA Y REDES DE INFORMACIÓN

BECERRA CAMACHO FERNANDO VINICIO
fernandobecerracv@gmail.com

DIRECTOR: ING. DAVID RAÚL MEJÍA NAVARRETE, MSc.
david.mejia@epn.edu.ec

CODIRECTOR: IVÁN MARCELO BERNAL CARRILLO, Ph.D.
Ivan.bernal@epn.edu.ec

Quito, Marzo 2014

DECLARACIÓN

Yo, Fernando Vinicio Becerra Camacho, declaro bajo juramento que el trabajo aquí descrito es de mi autoría; que no ha sido previamente presentado para ningún grado o calificación profesional; y, que he consultado las referencias bibliográficas que se incluyen en este documento.

A través de la presente declaración cedo mis derechos de propiedad intelectual correspondientes a este trabajo a la Escuela Politécnica Nacional, según lo establecido por la Ley de Propiedad Intelectual, por su Reglamento y por la normatividad institucional vigente.

Fernando Vinicio Becerra Camacho

CERTIFICACIÓN

Certifico que el presente trabajo fue desarrollado por Fernando Vinicio Becerra Camacho, bajo mi supervisión.

_____	_____
Ing. David Mejía MSc.	Iván Bernal Ph.D.
DIRECTOR DEL PROYECTO	CODIRECTOR DEL PROYECTO

AGRADECIMIENTO

Agradezco a Dios, a mis padres que me apoyaron durante todo estos años de estudio con su sabiduría, paciencia y comprensión que me ha ayudado a ser la persona que soy en este momento. A mi hermano, a mi abuelita que siempre me apoyaron en todo y a toda mi familia que ha estado conmigo.

Agradezco al Iván Bernal Ph.D. por su valiosa y desinteresada colaboración durante la elaboración de este proyecto.

Al Ing. David Mejía MSc. por su apoyo incondicional y paciencia durante el desarrollo de este proyecto.

A todos mis amigos de Redes entre ellos: Ely, Gaby, David, Fausto, Pato, José, Andrea con quienes he compartido las horas más valiosas durante mi vida estudiantil y que me han respaldado en todo. A los Júnior de Boca, que a más de un equipo, fuimos una pequeña familia que estuvimos en los buenos y malos momentos, dentro y fuera de la Escuela Politécnica Nacional.

ÍNDICE

ÍNDICE DE FIGURAS

ÍNDICE DE CÓDIGOS

ÍNDICE DE TABLAS

RESUMEN

En diciembre de 2011, la Escuela Politécnica Nacional fue declarada ganadora del Primer Concurso “Equipa tu universidad para el desarrollo de Aplicaciones de Televisión Digital Terrestre”, con lo cual se adquirió un compromiso de diseñar aplicaciones interactivas en diferentes áreas de interés público. Por este motivo se ha definido para este proyecto el desarrollo de aplicaciones interactivas relacionadas con la temática del Sistema de Educación Superior.

Las aplicaciones de interactividad en Televisión Digital, en unos casos, amplían y complementan la información sobre el tema que trata el programa de televisión que se está mirando, y en otros, se presentan como un contenido de interactividad sobre algún tema de interés general. La interactividad presenta al usuario un escenario con facilidades muy útiles a las cuales puede acceder cuando éste lo desee, todo esto sin interrumpir la visualización del programa de televisión y utilizando las teclas del control remoto.

El desarrollo de Aplicaciones para Televisión Digital intenta hacer uso de los recursos disponibles en Televisión Digital para proveer contenido interactivo que permita capacitar e informar a la población en temas de interés general.

En los últimos años se han venido realizando cambios fundamentales en el Sistema de Educación Superior. La población en general debe estar informada de dichos cambios. Entre estos cambios se pueden mencionar: la categorización de las universidades en el Ecuador la cual divide en diferentes categorías a todas las universidades tanto públicas como privadas; el gobierno oferta actualmente becas en las mejores universidades del mundo, las cuales son desconocidas por los posibles beneficiarios; un nuevo sistema de ingreso a las universidades está a cargo del SNNA (Sistema Nacional de Nivelación y Admisión) y Yachay la ciudad del futuro, la cual debería encaminar a Ecuador a cambiar la matriz productiva. Todos los puntos antes mencionados deben ser informados a los ciudadanos ecuatorianos, para lo

cual se desarrolló aplicaciones interactivas para televisión digital porque este medio es un masivo y de un gran alcance.

Además de esto se realizó encuestas con todas las aplicaciones para obtener una calificación de los usuarios que ocuparon las aplicaciones, los resultados de estas encuestas se utilizaron para mejorar las aplicaciones.

Adicionalmente se desarrolló un *plugin* para la herramienta *Composer* NCL, el cual está enfocado en la obtención de información de un *feed* RSS que se lo va a presentar en una aplicación para televisión digital.

PRESENTACIÓN

El desarrollo de este Proyecto se presenta en cuatro capítulos como se explica a continuación:

En el Capítulo 1 se realiza un breve resumen sobre el marco teórico de Televisión Digital, se describe brevemente el *middleware* Ginga, se presentan las opciones de desarrollo para aplicaciones interactivas que el *middleware* ofrece y se realiza una breve explicación sobre NCL y LUA. También se plantea realizar una revisión de las guías para la realización de aplicaciones interactivas.

En el Capítulo 2 se plantea el diseño y se implementa un *plugin* para el IDE de desarrollo de aplicaciones interactivas *Composer* NCL. El *plugin* permitirá usar *feed* RSS para obtener noticias desde sitios que el usuario especifique y se realizan las pruebas respectivas del *plugin*.

En el Capítulo 3 se efectúa el análisis de los requerimientos de las aplicaciones interactivas a implementarse. Se desarrollan cuatro aplicaciones interactivas enfocadas en la temática de la educación superior y se sincroniza una de ellas con el video. En este capítulo se especifican las pruebas realizadas con el *plugin* desarrollado, implementando lectores RSS específicos para las aplicaciones desarrolladas. Además, se realizan pruebas de las aplicaciones interactivas tanto en simuladores *set-top box* y en equipos *set-top box* disponibles. Finalmente, se procede a realizar encuestas MOS para las aplicaciones desarrolladas; la información obtenida mediante las encuestas permite realimentar las aplicaciones con la finalidad de mejorarlas.

En el Capítulo 4 se establecen las conclusiones y recomendaciones obtenidas en el desarrollo y aplicación del presente proyecto de titulación, orientando los resultados y beneficios conseguidos en el mismo.

En los Anexos se agrega el código de todas las aplicaciones realizadas y del *plugin* consumidorRSS también se encuentra el manual de AVALPA.

CAPÍTULO 1

FUNDAMENTO TEÓRICO

1.1 TELEVISIÓN DIGITAL TERRESTRE [1], [2]

La televisión digital terrestre ha provocado cambios significativos en el ámbito tecnológico, en la producción de programas y en los servicios con un valor añadido a la programación normal que se ofrece al espectador respecto a la televisión analógica.

En la parte tecnológica, se pueden transmitir una o varias señales de televisión digital en el mismo ancho de banda que se emplea en la televisión analógica de 6 MHz gracias a la utilización de técnicas de compresión de las señales de imagen y de sonido. En la televisión digital se tiene capacidad para un número variable de canales en función de la velocidad de transmisión, pudiendo oscilar entre un único canal de alta definición a cuatro canales con calidad de definición estándar de similares características a las de un canal de televisión analógico o incluso más canales, con calidad inferior a la definición estándar que se presenta actualmente.

En la televisión analógica no se aprovecha en la totalidad el ancho de banda, con el aumento de canales de televisión analógica la interferencia que se genera entre ellos se convierte en un problema grave ya que no se ve con nitidez la programación de cada de canal y se introduce la programación de un canal vecino, esto es incómodo para el televidente. La televisión digital tiene una mejor calidad tanto de imagen como de sonido, debido a que sus señales tienen un nivel superior de protección frente al ruido, un mejor aprovechamiento del ancho de banda, entre otras.

Un beneficio adicional cuando se transmite la señal de televisión digital es que junto a la transmisión de información convencional se puede transmitir información interactiva que va a complementar el contenido audiovisual.

En la producción de programas para televisión digital se puede introducir la interactividad en este medio que actualmente es pasivo; mediante aplicaciones se puede disponer de servicios interactivos que permiten que el televidente tenga un rol más participativo.

Además, con la interactividad se puede implementar servicios con valor agregado y que pueden cambiar el estilo de realizar los negocios en televisión, se pueden realizar aplicaciones interactivas que informen u ofrezcan un producto determinado. Esto conlleva a la convergencia de varios sectores como son: diseñadores gráficos, comunicadores sociales y programadores para realizar aplicaciones que sean completas tanto visualmente, como en contenidos y estructura.

1.1.1 CARACTERÍSTICAS [2], [3], [4], [5], [6]

1.1.1.1 Televisión de alta definición

La señal de televisión de alta definición (HD) se intentó transmitir por la televisión analógica pero se presentaron inconvenientes, el más delicado de todos era que para transmitir un solo canal de televisión analógica en alta definición se debía tener un ancho de banda igual a tres canales de definición estándar (SD), este inconveniente frenó su desarrollo, pero con las innovaciones de la transmisión de televisión digital se logró transmitir un canal de alta definición optimizando el uso del ancho de banda.

Las diferencias entre la televisión estándar y la televisión de alta definición radican en tres aspectos fundamentales que son: el incremento de la resolución de la imagen, la relación de aspecto y la capacidad de audio multicanal.

La resolución está basada en el número de píxeles que se utilizan para formar una imagen en la pantalla, las resoluciones de alta definición en televisión digital son 1080p[1] y 720p como se observa en la Tabla 1.1. La televisión de alta definición tiene

[1] 1080 representa 1080 líneas horizontales de resolución vertical de pantalla, mientras que la letra “p” significa *progressive scan* que es un método de exploración secuencial de las líneas de una imagen de televisión.

una resolución básica de 1280x720 píxeles que da como resultado 921.600 píxeles para una imagen.

	Definición	Píxeles [píxel]
Televisión SD	480i	640x480
Televisión HD	720p	1280x720
Televisión HD	1080p	1920x1080

Tabla 1.1 Tipos de definición de televisión

La relación de aspecto es la proporción entre el ancho y la altura de la imagen transmitida; en la Figura 1.1 se observan las diferencias físicas entre las definiciones de la televisión, una imagen para que sea de alta definición debe cumplir con la relación de aspecto 16:9 que corresponde a las definiciones de 720p y 1080p.

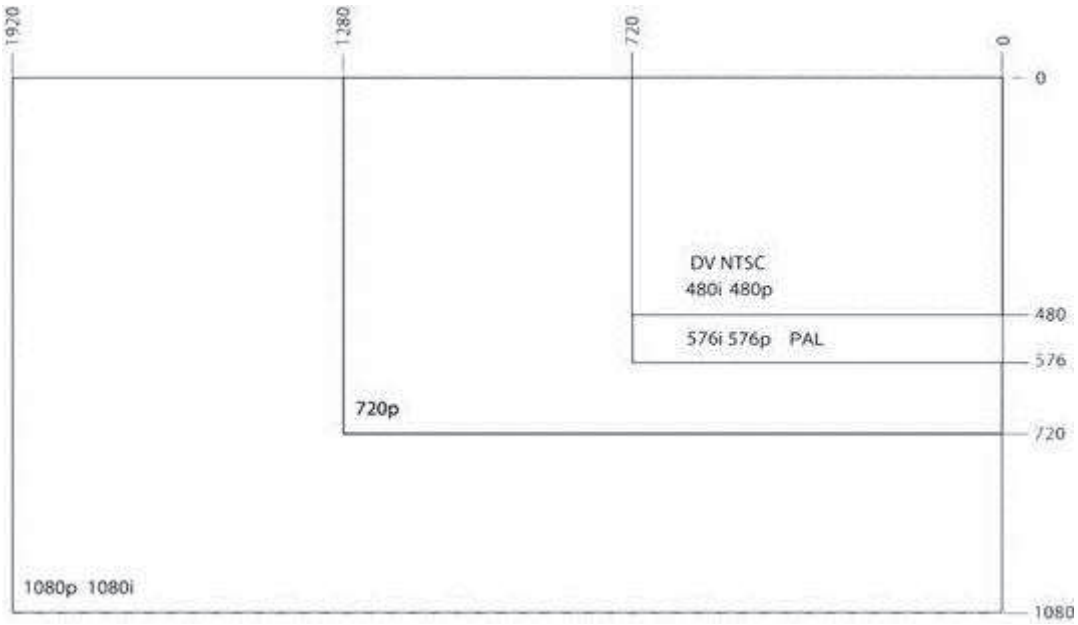

Figura 1.1 Relación de aspecto en televisión [6]

El audio multicanal permite la inclusión de varios canales de audio de forma simultánea. Desde la aparición del formato estéreo (dos canales), la evolución de la calidad del sonido había sufrido pequeñas mejoras debido a las limitaciones de los formatos analógicos, pero el formato digital ha proporcionado una mejor calidad del audio, permitiendo la aparición de 6 o más canales simultáneos.

1.1.1.2 Transmisión multicasting

La transmisión digital utiliza de forma eficiente el espectro radioeléctrico ya que en el ancho de banda que se transmite un canal de televisión analógica se puede transmitir más de cuatro canales con una definición estándar pero de forma digital o un canal en alta definición. Los cuatro canales se pueden transmitir de forma simultánea con todas las señales (audio y video de cada canal), a esto se lo denomina multiprogramación.

1.1.1.3 Transmisión datacasting

La posibilidad de transmitir datos con la señal de la televisión digital se conoce con el nombre de *datacasting*, es esta característica la que crea la posibilidad de implementar aplicaciones con un valor añadido.

Con estas aplicaciones se puede ganar interactividad, como por ejemplo, por medio de la televisión se puede obtener información de noticias, estados climáticos, información sobre un comercial en específico o sobre la programación normal de un canal en un día determinado.

1.1.1.4 Video mejorado

El procesamiento digital del video da la posibilidad de transmitir una imagen más detallada y con menor grado de error. Con estas características se optimiza el video enviando imágenes con gran resolución y que en conjunto con el *datacasting* permite disponer de una transmisión interactiva.

1.2 VENTAJAS DE LA TELEVISIÓN DIGITAL SOBRE LA TELEVISIÓN ANALÓGICA [7], [2], [8], [5]

Como se mencionó anteriormente la televisión digital permite incrementar el número de canales, también se disminuye el costo de distribución porque la transmisión de televisión digital requiere menor potencia que la analógica y que un solo productor de televisión ahora puede disponer de varios canales; en función de lo mencionado

anteriormente, se puede transmitir en diferentes definiciones lo que permitirá transmitir en el ancho de banda asignado al canal todas las programaciones de acuerdo a lo que el televidente haya seleccionado.

Las señales enviadas en forma digital resisten de mejor manera la transmisión multitrayecto y además son más robustas frente al ruido y las interferencias. La imagen ya no presenta distorsiones del tipo doble imagen o efecto "nieve" como sucede en la televisión analógica.

Una característica importante de la transmisión de televisión digital es la codificación, ésta dispone de mecanismos para la detección y corrección de errores que disminuye la tasa de error en las señales recibidas en entornos especialmente desfavorables, la televisión analógica no tiene ninguno de estos mecanismos.

La transmisión de televisión digital permite una optimización del espectro radioeléctrico puesto que con su diseño es posible usar todos los canales de la banda, sin necesidad de dejar canales de guarda para reducir las interferencias, siendo esto una gran ventaja frente a la televisión analógica en la que se debe asignar un canal para que no exista interferencia entre ellos.

La transmisión digital permite la recepción móvil de la televisión. Además, permite una reducción en el tamaño de las antenas receptoras, lo que posibilita su recepción en terminales más pequeños como pueden ser un PDA[2] o un teléfono celular.

1.2.1 ESTÁNDARES DE TELEVISIÓN DIGITAL TERRESTRE [9]

Los estándares de transmisión de televisión digital se han adoptado en diferentes países de acuerdo a las características técnicas y servicios que ofrecen. En la Figura 1.2 se puede observar cómo están distribuidos en el mundo estos estándares.

Los estándares a nivel mundial de televisión digital terrestre son:

- *Advanced Television Systems Committee* (ATSC) [10]

[2] PDA *(Personal Digital Assistant)* es una agenda de electrónica.

- *Digital Video Broadcasting* (DVB) [11]
- *Digital Terrestrial/Television Multimedia Broadcasting* (DTMB) [12]
- *Integrated Services Digital Broadcasting* (ISDB-T) [1]

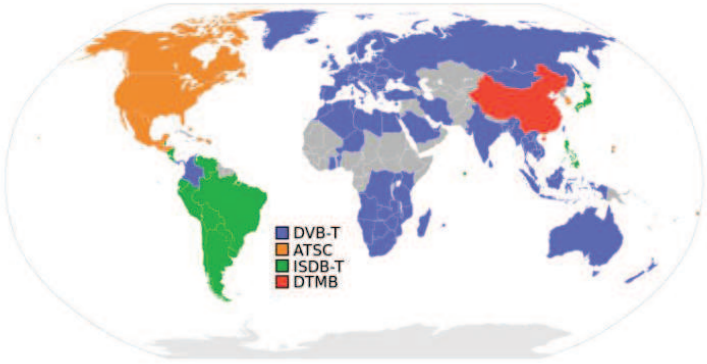

Figura 1.2 Estándares de televisión digital en el mundo [9]

1.2.2 LA TELEVISIÓN DIGITAL TERRESTRE EN EL ECUADOR [5]

En Ecuador se adoptó el estándar ISDB-Tb, el 26 de marzo del 2010, como norma de televisión digital terrestre abierta. En Ecuador se especificó que el 31 de diciembre del 2018 se va a realizar el apagón analógico, el cual consta de tres fases según la resolución RTV-681-24-CONATEL-2012, en la Figura 1.3 están detalladas las fases de dicho apagón.

TDT ECUADOR

FECHAS APAGÓN ANALÓGICO

FASES	CIUDADES	FECHA
Fase 1	Áreas de cobertura de las estaciones que al menos cubran una capital de provincia, cabecera cantonal o parroquia con población mayor a 500.000 habitantes	31 de diciembre de 2016
Fase 2	Áreas de cobertura de las estaciones que al menos cubran una capital de provincia, cabecera cantonal o parroquia con población entre 500.000 y 200.000 habitantes	31 de diciembre de 2017
Fase 3	Áreas de cobertura de las estaciones que al menos cubran una capital de provincia, cabecera cantonal o parroquia con población menor a 200.000 habitantes	31 de diciembre de 2018

Figura 1.3 Apagón analógico en el Ecuador [5]

1.2.2.1 Integrated Services Digital Broadcasting versión Brasileña (ISDB-Tb) [1]

El estándar de televisión digital brasileño (ISDB-Tb) está basado en el estándar japonés (ISDB-T), sus diferencias con este estándar son el uso de tecnologías de compresión de video MPEG-4 (H.264[3]) y de audio HE-AAC[4](para *one seg* el audio se comprime en HE-AAC v.1 de baja complejidad y para el video es el mismo tipo de compresión), mientras que el estándar japonés utiliza para compresión de video MPEG-2 y de audio MPEG-2 ACC[5], además, el estándar brasileño tiene un *middleware* totalmente innovador llamado Ginga y en el estándar japonés se utiliza BML[6]. La modulación en los dos sistemas es idéntica, al igual que el transporte que se realiza con el estándar MPEG-2. En la Tabla 1.2 se resume la información de ambas normas.

Tecnología de Transmisión	**ISDB-T**	**ISDB-Tb**
Aplicativos	Interactivos	Interactivos
Middleware	BML	Ginga
Compresión de audio	MPEG-2 AAC	HE-AAC
Compresión de video	MPEG-2	MPEG-4 (H.264)
Transporte	MPEG-2	MPEG-2
Transmisión y Modulación	BST-OFDM[7]	BST-OFDM

Tabla 1.2 Diferencias entre los estándares ISDB-T y ISDB-Tb

[3] H.264 es una norma que define un códec de vídeo de alta compresión, desarrollada conjuntamente por el ITU-T VCEG y el ISO/IEC MPEG.
[4] HE-AAC (*High-Efficiency Advanced Audio Coding*) es un formato de compresión de audio digital con pérdidas definido como un perfil MPEG-4.
[5] MPEG-2 AAC *(Advanced Audio Coding)* es un formato de audio basado en un algoritmo de compresión con pérdidas, un proceso por el que se eliminan algunos de los datos para poder obtener el mayor grado de compresión.
[6] BML *(Broadcast Markup Language) middleware* del estándar japonés
[7] BST-OFDM *(Band Segmented Transmission-OFDM)* tiene la misma técnica que OFDM pero el canal se divide en trece segmentos.

1.3 INTERACTIVIDAD EN TELEVISIÓN DIGITAL

La interactividad, en el contexto de TDT, es la capacidad de transformar un medio pasivo en un medio dinámico, esto es posible por medio de aplicaciones que complementan los contenidos de la televisión, estas aplicaciones pueden brindar información adicional como programación de los canales, información asociada al contenido audiovisual, participar en concursos solo utilizando el control remoto, etc., toda esta información lo puede revisar el televidente cuando así lo desee.

1.3.1 APLICACIONES INTERACTIVAS

Las aplicaciones interactivas de televisión digital son programas que tienen la capacidad de brindar contenidos adicionales a la programación normal y permiten obtener información extra a voluntad de los televidentes, pudiendo ellos escoger los contenidos que deseen.

1.3.2 CLASIFICACIÓN DE APLICACIONES INTERACTIVAS

De acuerdo con la existencia o no del canal de retorno, los niveles de interactividad son clasificados en:

Interactividad Local: No utiliza el canal de retorno. Pueden realizarse interacciones como: configuración de leyendas, juegos residentes y acceso a la guía de programación electrónica.

Interactividad Remota: Utiliza el canal de retorno. Pueden realizarse interacciones como: comercio electrónico, acceso a cuentas bancarias, servicios de salud, aplicaciones para educación a distancia, etc.

La interactividad remota puede ser de dos tipos que son: unidireccional y bidireccional.

Unidireccional: permite al STB[8] solo el envío o recepción de datos, por ejemplo: la compra de un determinado producto, votación e investigación de opinión.

Bidireccional: permite al STB el envío y recepción de datos, por ejemplo se tiene un sistema de votación que muestra los resultados además que también se puede votar.

Las aplicaciones interactivas se pueden clasificar por su contenido, como se observa en las siguientes clases:

EPG: El EPG (*Electronic Program Guide*) es un servicio de guía para contenidos de programas de televisión; por ejemplo, el televidente podrá navegar por un conjunto de programas y servicios ofrecidos y escoger lo que más le agrade.

T-Gobierno: Son aplicativos enfocados a servicios gubernamentales. Ofrece servicios importantes evitando el desplazamiento por parte de los televidentes hacia las oficinas, ahora las consultas son realizadas a través del STB.

T-Salud: Los televidentes pueden recibir información relativa a campañas de salud e incluso ponerse en contacto con los servicios con el objetivo de solicitar información.

T-Contenidos: Parte de los contenidos actualmente disponibles en Internet pueden ser ofrecidos a través de los servicios interactivos de televisión digital teniendo en cuenta las posibilidades y limitaciones tecnológicas.

T-Marketing: Permite la integración de publicidad interactiva en la televisión digital, ofreciendo información adicional a la presentada en la propaganda emitida y permitiendo la participación de los televidentes en campañas publicitarias, pudiendo ofrecerse también la posibilidad de comprar productos.

T-Comunidad: Permite establecer un canal de comunicación e interacción entre los telespectadores y los contenidos de la televisión mediante la participación en aplicaciones, juegos y concursos, así como interactuando con otros telespectadores.

[8] *Set-top box:* receptor o decodificador de televisión digital.

T-Formación: Los servicios educativos son uno de los principales pilares de las aplicaciones interactivas en la televisión digital porque en la mayoría de hogares se tiene acceso a un televisor y no a una computadora, por esta razón la televisión es un medio masivo de difusión para el servicio educativo, capaz de reducir la brecha digital entre ciudadanos. La televisión digital como medio de acceso sencillo y familiar potencialmente disponible en un gran número de entornos supone una oportunidad nueva para modernizar los sistemas de educación y formación.

1.4 CANAL DE RETORNO [13]

El canal de retorno es un medio de transmisión que permite la comunicación entre el receptor de televisión con el operador de servicio interactivo, y facilita el envío y recepción de datos que el usuario intercambia con el operador para poder hacer uso de ciertos servicios como se puede observar en la Figura 1.4.

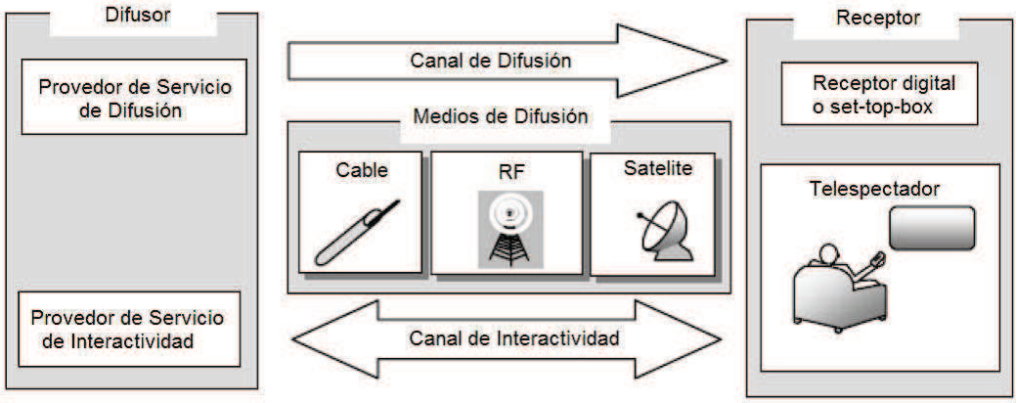

Figura 1.4 Esquema de funcionamiento de televisión digital [13]

Este canal de retorno puede ser implementado mediante varias tecnologías existentes en el mercado, cada una de las cuales presentan características que son apropiadas para determinadas zonas, regiones o países. Como se ha mencionado anteriormente, una de las ventajas más importantes de la televisión digital es que el televidente pueda interactuar con la televisión a través del control remoto, permitiéndole ver información adicional, manipular la programación de los canales de televisión, comprar servicios o productos, participar en concursos, votaciones, etc.

Sin embargo, este tipo de aplicaciones, además de necesitar una plataforma para aplicaciones interactivas, requiere de una comunicación, es decir necesita que el canal de retorno permita enviar y recibir datos para que así el televidente tenga la capacidad de interactuar directamente con un servidor remoto.

El contenido televisivo puede estar formado por flujos de audio, video o datos que son transmitidos por estaciones de televisión y recibidos por televisiones que soporten transmisión digital o por los STB que se encargan de la conversión de la señal digital para que pueda ser captada por televisores analógicos, además posee un canal de retorno para ofrecer la interactividad entre la televisor y el televidente.

1.4.1 TIPOS DE CANALES DE RETORNO [13], [12], [14]

A continuación se mencionan las tecnologías existentes para la implementación del canal de retorno, estas tecnologías están especificadas en [21], y son las siguientes:

- ISDN[9]
- Ethernet (ADSL[10],FTTH[11],DOCSIS[12])
- Wi-Fi[13]
- WiMAX[14]
- GSM[15]

[9] ISDN (*Integrated Services Digital Network*) Es sistema para las conexiones de teléfonos digitales, especialmente creado para proveer servicios como el envío de voz, de video, así como datos.
[10] ADSL (*Asymmetric Digital Subscriber Line*) Consiste en una transmisión analógica de datos digitales apoyada en la línea telefónica convencional o línea de abonado.
[11] FTTH (*Fiber To The Home*) enmarcada dentro de las tecnologías FTTx, se basa en la utilización de cables de fibra óptica adaptados a esta tecnología para la distribución de servicios avanzados, como telefonía, Internet de banda ancha y televisión.
[12] DOCSIS (*Data Over Cable Service Interface Specification*) es un interfaz de comunicaciones y operaciones para los datos sobre sistemas de cable, lo que permite añadir transferencias de datos de alta velocidad a un sistema de televisión por cable.
[13] Wi-Fi es un mecanismo de conexión de dispositivos electrónicos de forma inalámbrica, definido por el estándar IEEE 802.11.
[14] WiMAX (*Worldwide Interoperability for Microwave Access*) es una norma de transmisión de datos que utiliza las ondas de radio y pueden tener grandes coberturas, el estándar que define esta tecnología es el IEEE 802.16.
[15] GSM (*Global System for Mobile*) originalmente como estándar Europeo abierto para que una red digital de teléfono móvil que soporte voz, datos, mensajes de texto y *roaming* en varios países, se denomina una tecnología de segunda generación.

- CDMA[16]
- Módems dial-up[17]

En la Figura 1.5 se puede observar a un *set-top box* que está utilizando el canal de retorno.

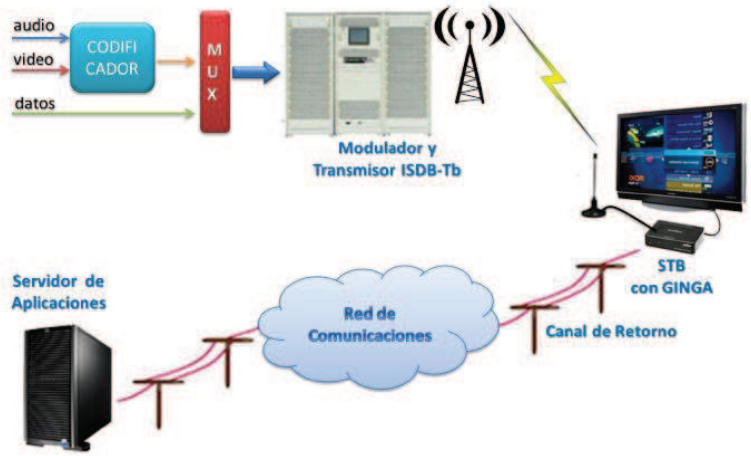

Figura 1.5 *Set top box* con canal de retorno [12]

1.5 MIDDLEWARE GINGA [8]

Para el estándar ISDB-Tb se definió un *middleware* propio denominado Ginga. En la Figura 1.6 se muestra el modelo de referencia de ISDB-Tb. Este *middleware* fue desarrollado por el laboratorio de Telemidia de la PUC-Rio (Pontifícia Universidade Católica do Río de Janeiro) y LAViD (Laboratório de Aplicações de Vídeo Digital) de la UFDP (Universidade Federal da Paraiba). El nombre fue escogido en reconocimiento a la cultura, arte y continua lucha por la libertad e igualdad del pueblo brasileño. Ginga permite presentar los contenidos independientemente de la plataforma de hardware del fabricante y el tipo de STB.

Las aplicaciones ejecutadas sobre Ginga son clasificadas en dos categorías dependiendo del tipo de lenguaje de programación que se utilice. Las aplicaciones

[16] CDMA (*Code Division Multiple Access*) permite que múltiples terminales compartan el mismo canal de frecuencia, identificándose el canal de cada usuario mediante códigos.
[17] Módems dial-up es una forma económica de acceso a Internet en la que el cliente utiliza un módem para llamar a través de la Red Telefónica Conmutada al nodo del ISP.

procedimentales son escritas usando el lenguaje Java, utilizan una estructura llamada GINGA-J y las aplicaciones declarativas son escritas usando el lenguaje NCL, utilizan una estructura llamada GINGA-NCL.

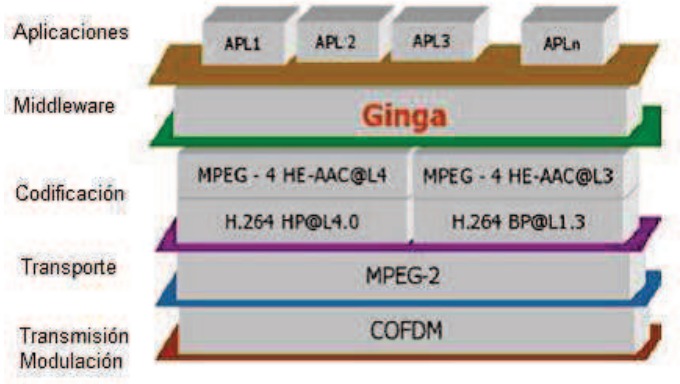

Figura 1.6 Modelo de referencia de televisión digital [8]

La arquitectura de referencia del *middleware* Ginga puede ser dividida en tres grandes estructuras: GINGA-CC (*Common Core*), el ambiente de declarativo GINGA-NCL y el ambiente de procedimiento GINGA-J.

1.5.1 NÚCLEO COMÚN DE GINGA (GINGA-CC)

El núcleo común de Ginga tiene servicios necesarios tanto para el motor de presentación (declarativo) como para el motor de ejecución (procedimiento). Esta estructura es la interfaz directa con el sistema operativo proporcionando un puente estrecho con el hardware. En esta estructura se accede al sintonizador de canales, sistema de archivos, terminal gráfico, entre otros. Está compuesto por los decodificadores de contenido común y por procedimientos para obtener contenidos transportados en flujos MPEG-2. Los decodificadores de contenido común sirven tanto para las aplicaciones de procedimiento así como para las aplicaciones declarativas, las cuales requieren decodificar para presentar tipos comunes de contenidos como PNG[18], JPEG[19], entre otros formatos. El núcleo común de Ginga

[18] PNG (*Portable Network Graphics*) es un formato gráfico basado en un algoritmo de compresión sin pérdida para bitmaps no sujeto a patentes.
[19] JPEG (*Joint Photographic Experts Group*) es comité de expertos que creó un estándar de compresión y codificación de archivos de imágenes fijas.

que debe obligatoriamente ser compatible con el modelo conceptual de exhibición se define en [22]. En la Figura 1.7 se muestra la estructura de GINGA-CC

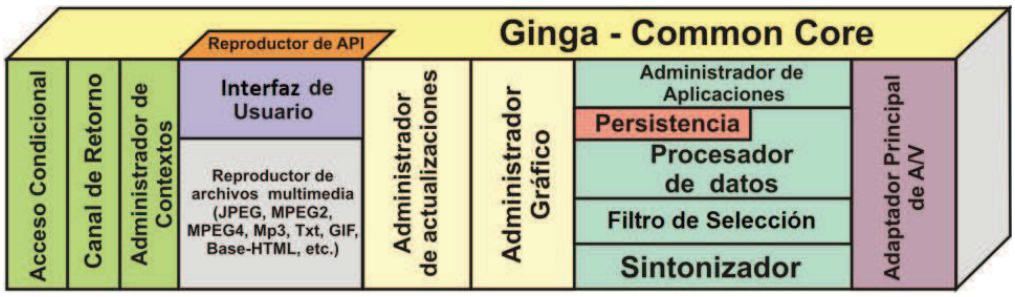

Figura 1.7 Estructura GINGA-CC [8]

A continuación se presentan los elementos que conforman GINGA-CC:

Sintonizador: Este componente es responsable de sintonizar un canal, eligiendo un canal físico y los flujos de transporte que están siendo enviados por el mismo.

Filtro de Selección: Este componente es capaz de buscar en el flujo la parte exacta que las API necesitan para su ejecución.

Procesador de Datos: Es el componente responsable de acceder, procesar y transferir los datos recibidos por la capa física. También es responsable de notificar a los otros componentes sobre cualquier evento que se ha recibido.

Persistencia: Este componente está capacitado para guardar archivos, incluso después que han sido utilizados para que puedan ser abiertos en otra ocasión.

Administrador de Aplicaciones: Este componente es responsable de cargar, configurar, inicializar y ejecutar cualquier aplicación ya sea declarativa o procedural. También es responsable de controlar el ciclo de vida de las aplicaciones, eliminarlas cuando sea necesario, además de controlar los recursos utilizados por las API.

Adaptador Principal de A/V: En este componente, las aplicaciones consiguen presentar el flujo de audio y vídeo. Esto es necesario cuando una aplicación necesita controlar sus acciones de acuerdo con lo que está recibiendo del STB.

Administrador de Gráficos: Las normas del *middleware* definen como se presentan al usuario las imágenes, videos, datos, etc., administrando las presentaciones de la misma manera que está definida en [23].

Administrador de Actualizaciones: Es el componente que gestiona las actualizaciones del *middleware*.

Reproductor de Archivos Multimedia: Este componente tiene las herramientas necesarias para presentar los archivos multimedia.

Interfaz de Usuario: Este componente es responsable de captar e interpretar los eventos generados por los usuarios, como comandos del control remoto y notificar a los módulos interesados.

Administrador de Contextos: Este componente es el responsable de captar las preferencias del usuario notificando a los otros componentes interesados.

Canal de Retorno: Este componente proporciona la interfaz de las capas superiores con el canal retorno gestionándolo de modo que los datos sean transmitidos cuando el canal esté disponible.

Acceso Condicional: Este componente está encargado de restringir contenidos inapropiados recibidos por los canales.

1.5.2 GINGA-NCL

La estructura GINGA-NCL provee una infraestructura de presentación para aplicaciones declarativas escritas en el lenguaje NCL (*Nested Context Languaje*) que tiene una estructura XML[20] (*eXtensible Markup Language*), con facilidades para los aspectos de interactividad, sincronismo, espacio-temporal entre objetos media, adaptabilidad, soporte a múltiples dispositivos y soporte a la producción de programas interactivos en vivo no-lineales. En la Figura 1.8 se observa el módulo GINGA-NCL. La especificación de esta estructura se describe en [24] y [25].

[20] XML es un lenguaje de marcas desarrollado por el *World Wide Web Consortium* (W3C) utilizado para almacenar datos en forma legible.

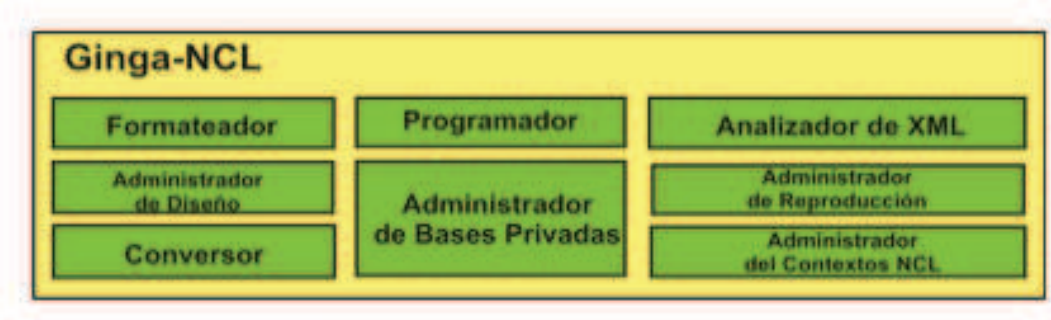

Figura 1.8 Estructura GINGA-NCL [8]

A continuación se presentan los elementos que conforman el módulo GINGA-NCL:

Formateador: Se encarga de recibir y controlar las aplicaciones multimedia escritas en NCL que son entregadas por GINGA-CC.

Analizador de XML y Conversor: Realizan la conversión de la aplicación NCL en la estructura interna de datos de GINGA-NCL para controlar la aplicación, estos componentes son solicitados por el "Formateador".

Programador: Permite organizar el orden de la presentación del documento NCL. Este componente es responsable de dar la orden para iniciar la reproducción apropiada del tipo de contenido media para exhibirlo en el momento indicado.

Base Privada: El "Motor de Presentación" interactúa con un conjunto de aplicaciones NCL que están dentro de una estructura conocida como "Base Privada".

Administrador de la Base Privada: Este componente se encarga de recibir los comandos de edición de los documentos NCL y también realiza el mantenimiento a los documentos presentados.

Administrador del Diseño: Este componente es responsable de mapear todas las regiones definidas en una aplicación NCL.

Administrador del Contexto NCL: Este componente soporta el contenido enviado desde Ginga-CC.

Administrador de Reproducción: Este componente se encarga de organizar el orden de presentación de una aplicación NCL.

1.5.3 GINGA-J [15]

GINGA-J es la estructura lógica del sistema Ginga que procesa el contenido de los objetos Xlet[21]. Un componente clave del ambiente de las aplicaciones procedimentales es el motor de ejecución de contenidos de procedimiento compuesto por la máquina virtual de Java. La especificación de esta estructura está basada en [26].

Para satisfacer los requerimientos específicos de Brasil y al mismo tiempo mantener la compatibilidad internacional con la API[22] de GEM[23], Ginga establece tres grupos de API llamados: Verde, Amarillo y Azul, como se observa en la Figura 1.9.

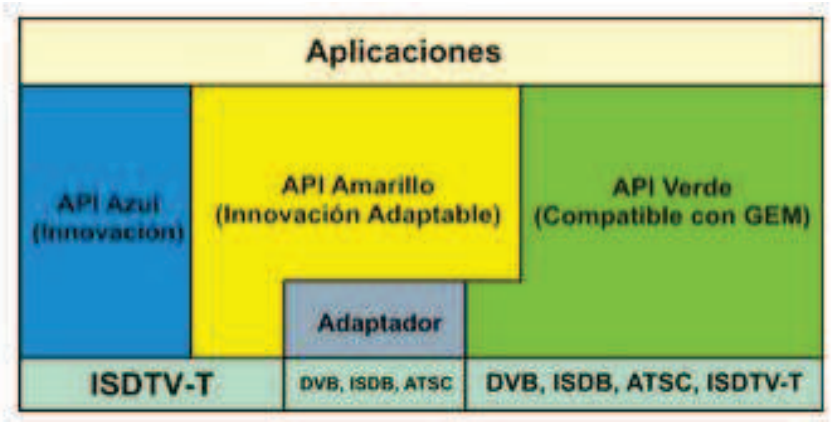

Figura 1.9 Estructura GINGA-J [8]

La API Verde es compatible con GEM, aquí se encuentran las API provenientes de los paquetes de JavaTV.

---

[21] Xlet es el nombre que reciben las aplicaciones tipo DVB-Java. Son desarrolladas en lenguaje de programación Java para la interfaz API MHP(*Multimedia Home Platform*), que define una plataforma común para las aplicaciones interactivas de televisión digital. Xlet está dirigido al desarrollo de aplicaciones para televisión digital dentro de la especificación Java TV.

[22] API (*Application Programming Interface*) Interfaz de programación de aplicaciones.

[23] GEM (*Globally Executable Multimedia Home Platform*) permitir que otros cuerpos de estandarización u organizaciones pudieran definir unas especificaciones basadas en el estándar MHP.

La API Amarillo está compuesta por la API JMF[24] 2.1, la misma que es necesaria para el desarrollo de aplicaciones con captura de sonido.

La API Azul permite al STB comunicarse con cualquier dispositivo con una interfaz compatible con el canal de retorno como Ethernet o PLC de red o inalámbrica como infrarrojos o Bluetooth. Además, aquí se encuentra la API que permite el desarrollo de las aplicaciones GINGA-J que tengan GINGA-NCL API puente.

1.6 NCL-LUA

1.6.1 NCL [14]

NCL (*Nested Context Language*) fue desarrollado en el Laboratorio TeleMídia de la PUC-Río de Janeiro, es el único lenguaje declarativo de dominio específico que satisface los siguientes requerimientos: cumplir con el sincronismo de una manera general, tener una interactividad con el usuario, definición de sincronismo temporal y espacial, separando la definición del contenido de los objetos de medios relacionados, adaptación de contenido, múltiples contenidos de exhibición y la edición en vivo. Todos estos requerimientos permiten el desarrollo de gran parte de las aplicaciones de televisión digital por personas no técnicas sin ningún conocimiento previo de programación.

NCL es un lenguaje con estructura XML (*eXtensible Markup Language*), que ofrece varias opciones para la creación de un documento hipermedia con las relaciones de sincronización entre sus componentes, está basado en el modelo conceptual NCM (*Nested Context Model*) que brinda una clara separación entre el contenido de medios de comunicación y la estructura de una aplicación. Un documento NCL define cómo los objetos están estructurados con los medios de comunicación y cómo están relacionados en el tiempo y en el espacio. El ambiente declarativo es en sí muy limitado porque no es extensible y no puede utilizar *scripts*.

[24] JMF (Java *Media Framework)* es una extensión de Java que permite la programación de tareas multimedia en este lenguaje de programación.

En NCL, un documento XHTML[25] es un tipo de elemento de los medios de comunicación de forma semejante a los lenguajes imperativos que pueden ser adicionados y utilizados como medios de comunicación. En concreto, el GINGA-NCL debe ofrecer soporte a dos lenguajes procedimentales como son Lua y Java. Lua es el lenguaje script de NCL, y Java debe seguir las especificaciones de GINGA-J.

1.6.1.1 Elementos de NCL

Los elementos que se utilizan para la programación en NCL son:

Regiones (*Regions*): Son áreas de presentación de elementos multimedia. Se definen en el encabezado (*head*) del documento NCL en la sección base de regiones (*regionBase*). Los documentos NCL poseen al menos una región que define la dimensión y las diferentes características de cómo se presentan uno o más elementos multimedia.

Descriptores (*Descriptors*): Son los elementos encargados de definir cómo será presentado un nodo multimedia, asociándolo a una región, están definidos en el encabezado (*head*) del archivo NCL en la sección llamada base de descriptores (*descriptorBase*).

Contextos (*Contexts*): Se los utiliza para estructurar un documento hipermedia, los mismos que pueden ser anidados con el objetivo de reflejar la estructura del documento y mejorar su organización.

Puertos (*Ports*): Forman puntos de interfaz de un contexto, ofreciendo acceso externo al contenido del mismo, es decir, para que un enlace apunte a un nodo interno éste debe poseer un puerto que lo dirija hacia dicho nodo.

Conectores (*Connectors*): Son elementos que definen uno o más roles para una condición de activación del enlace y para acciones que deben realizarse cuando el enlace esté activo.

---

[25] XHTML (*eXtensible HyperText Markup Language*) es básicamente HTML expresado como XML válido.

Enlaces (*Links*): Asocian nodos a través de conectores que definen la semántica de asociación entre ellos.

Enlazador (*Bind*): Este elemento permite asociar una interfaz de un objeto con un conector.

Switch: Un *switch* es un contexto con nodos alternativos, dentro del cual uno será activado. La decisión acerca de que nodo será activado es dada por reglas.

1.6.2 LUA [14], [15]

Lua es un lenguaje de extensión creado por miembros del Grupo de Tecnología en Computación Gráfica (Tecgraf) en la PUC-Rio, es suficientemente compacto para usarse en diferentes plataformas, sus variables no tienen tipo, los datos pueden ser lógicos, enteros, números de coma flotante o cadenas.

Lua es un lenguaje diseñado para apoyar la programación procedimental general con las descripciones de datos. También ofrece un buen soporte para programación orientada a objetos, programación funcional y programación orientada a datos. Lua está destinado a ser utilizado como un potente lenguaje ligero y también está creado como un lenguaje de *scripting*.

Siendo un lenguaje de extensión, Lua no tiene noción de programa principal (*main*), funciona embebido en un cliente anfitrión denominado programa contenedor o simplemente anfitrión (*host*). El contenedor puede invocar funciones para ejecutar un segmento de código Lua, puede escribir y leer variables de Lua y puede registrar funciones escritas en C para que sean llamadas por el código Lua a través del uso de dichas funciones, Lua puede ser ampliado para abarcar un amplio rango de diferentes dominios, creando entonces lenguajes de programación personalizados que comparten el mismo marco sintáctico.

Lua no comenzó con la televisión digital y no fue diseñado para ese fin, la televisión digital es un entorno más en el que este lenguaje demostró su fuerza y rendimiento.

1.7 COMPOSER NCL [16]

Composer NCL es una herramienta desarrollada por el Laboratorio TeleMídia del Departamento de Informática de la PUC-Rio. Con esta herramienta es posible construir programas audiovisuales interactivos con lenguaje NCL. Con esta herramienta se puede programar una aplicación interactiva sin tener conocimientos amplios sobre el lenguaje de programación NCL y además tiene muchas ayudas gráficas. En la Figura 1.10 se puede observar el software *Composer* NCL.

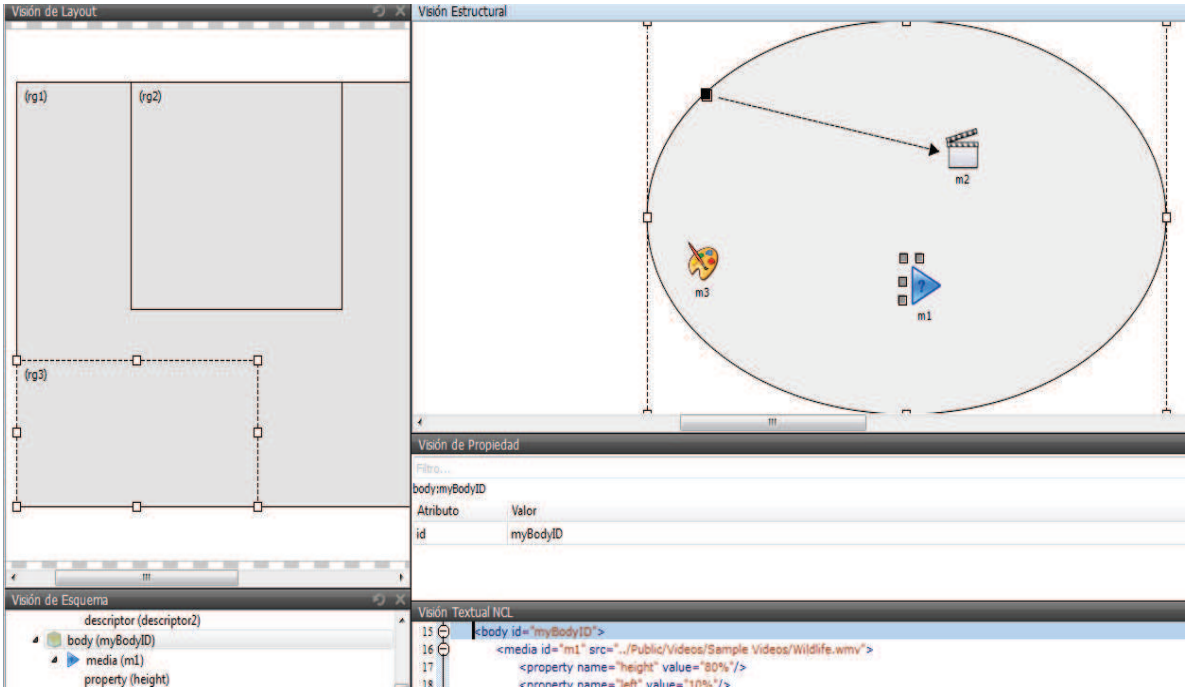

Figura 1.10 Software *Composer* NCL

El software *Composer* NCL está constituido por *plugins* los cuales ayudan al diseño de una aplicación interactiva, a continuación se detalla cada uno ellos.

1.7.1 VISTA DE LAYOUT

El *plugin* vista de l*ayout* presenta las regiones de pantalla en donde los elementos media pertenecientes al documento se presentarán. En la Figura 1.11 se muestra la vista de *Layout* de un documento hipermedia.

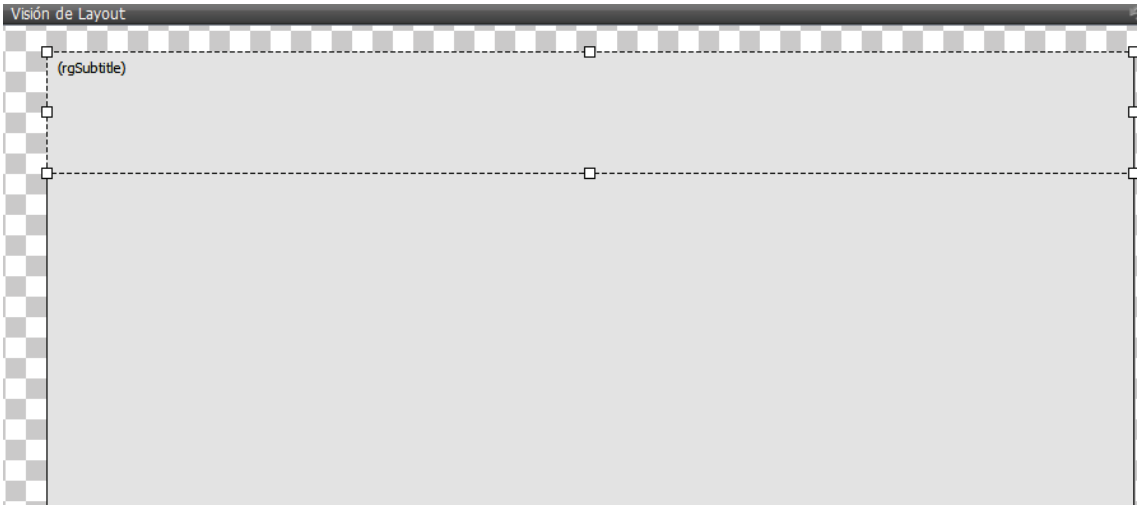

Figura 1.11 Vista de *Layout*

1.7.2 VISTA ESTRUCTURAL

El *plugin* vista estructural permite visualizar todos los elementos multimedia que contengan un documento NCL así como los enlaces, los conectores que interactúan entre los elementos, entre otros. En la Figura 1.12 se puede observar la vista estructural.

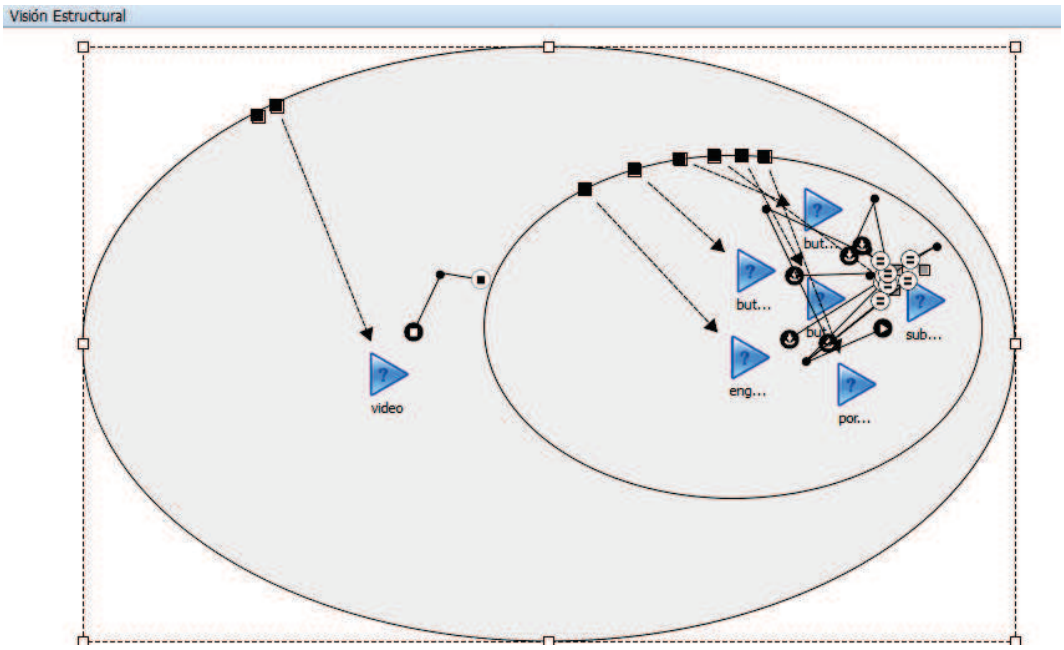

Figura 1.12 Vista estructural

1.7.3 VISTA TEXTUAL

El *plugin* vista textual muestra el código NCL tal como se observa en la Figura 1.13. Con este *plugin*, el usuario puede editar el código NCL como en un editor de texto, permitiéndole manipular todos los elementos NCL. Cuando se agregan códigos NCL en este *plugin* automáticamente se generan los elementos en los demás *plugins* sin tener que hacer operaciones adicionales.

Figura 1.13 Vista textual

1.7.4 VISTA DE ESQUEMA

El *plugin* vista de esquema presenta la estructura del documento de hipermedia como un árbol de datos como se observa en la Figura 1.14. Este tipo de vista es ideal para tener organizado todos los objetos que interactúan en una aplicación de televisión digital.

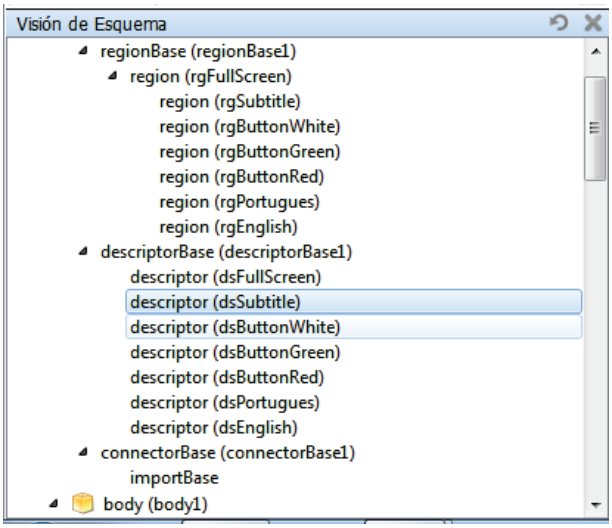

Figura 1.14 Vista de esquema

1.7.5 VISTA DE PROPIEDAD

El *plugin* vista de propiedad permite al usuario cambiar y observar las propiedades que tienen los elementos multimedia, como se puede ver en la Figura 1.15.

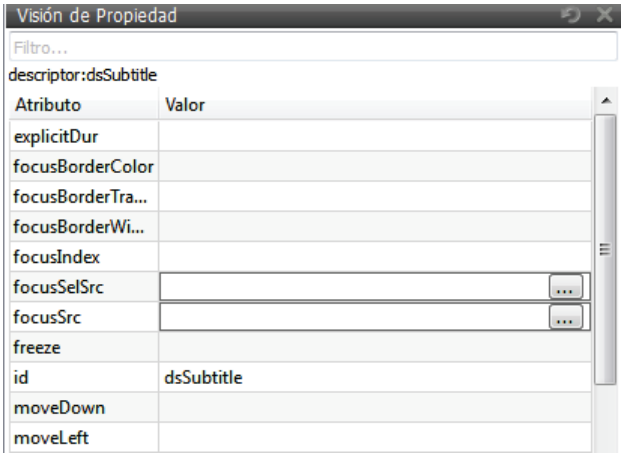

Figura 1.15 Vista de propiedad

1.7.6 PLUGIN VALIDADOR

El plugin validador ayuda a solucionar rápidamente los errores que se han cometido porque muestra el lugar donde se cometió el error y así poder depurarlo fácilmente. En la Figura 1.16 se muestran los errores cometidos en una aplicación NCL.

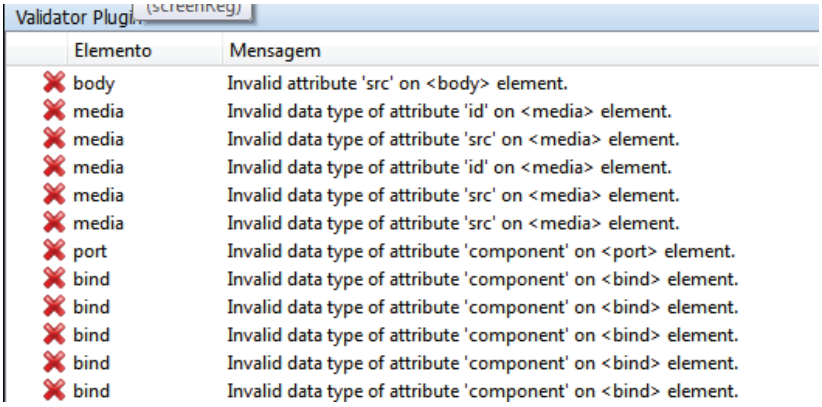

Figura 1.16 *Plugin* validador

1.8 CRITERIOS DE DISEÑO [20]

Para las aplicaciones Ginga se debe tener en cuenta que su diseño es totalmente diferente a diseñar una interfaz para usuarios de un computador, se deben cumplir muchos objetivos y puntos para que la aplicación tenga una buena acogida de los televidentes. A continuación se detallan los puntos que se deben tomar en cuenta para el desarrollo de las aplicaciones:

- Las aplicaciones deben tener interfaces gráficas que llamen la atención de los usuarios, estas interfaces deben ser innovadoras para que los usuarios las utilicen hasta por curiosidad.
- La aplicación no debe contener información extensa tanto de texto como de imágenes, para que los usuarios que están utilizando el televisor puedan relajarse ya que si existiese mucha información no les va a interesar.
- El texto a presentar debe ser resumido y solo se debe colocar lo más importante de cada punto.
- Es necesario analizar la combinación de los colores debido que una mala composición de los mismos puede causar una inadecuada opinión ante los televidentes o no pueden leer el texto expuesto.
- Cuando exista redimensionamiento de video se debe tener en cuenta la relación de aspecto que se está transmitiendo que es 16:9.

- El tamaño del redimensionamiento debe ser adecuado ya que muchas empresas pagan por el uso de espacios publicitarios y los canales de televisión no quieren perder esta fuente de ingresos.
- El uso de las aplicaciones debe ser lo más sencillo posible para que todas las personas las puedan utilizar.
- Cuando se tenga que presionar botones del control remoto se debe crear un menú para que se pueda entender el funcionamiento de la aplicación.
- El texto que va a ingresar debe ser de tamaño apropiado y de una fuente clara para poder observar el contenido desde una distancia adecuada del usuario.
- La aplicación debe permitir al usuario salir desde cualquier página en la que se encuentre para presentar la programación normal de la televisión.
- Se debe diseñar tomando en cuenta que existen diferentes tipos de resoluciones y configuraciones de pantalla.

1.8.1 CONTROL REMOTO

En el control remoto de la Figura 1.17 se pueden apreciar algunos botones que son comunes entre todos los controles que existen, por ejemplo para una aplicación se pueden utilizar las siguientes alternativas: los botones con los cursores de desplazamientos se pueden utilizar para desplazarse por los elementos de un menú; el botón “OK” serviría para seleccionar un elemento; el botón “EXIT” permitiría salir de la aplicación y los botones con los colores rojo, verde, azul y amarillo podrían usarse para acceder a funciones específicas de una aplicación.

Figura 1.17 Control remoto

REFERENCIAS BIBLIOGRÁFICAS

[1] PÁGINA OFICIAL DE ISDB-T. *http://www.dibeg.org/techp/techp.html*. Consultado en Enero 2013.

[2] FORO OFICIAL DE SBTVD. *http://forumsbtvd.org.br*. Consultado en Enero del 2013.

[3] Cubero, Manuel. *La Televisión Digital: Fundamentos y teorías*. 2009.

[4] Benoit, Herve. *Digital Television: Satellite, Cable, Terrestrial, IPTV, Mobile TV in the DVB Framework*. 2008.

[5] SUPERTEL. *INFORME PARA LA DEFINICIÓN E IMPLEMENTACIÓN DE LA TELEVISIÓN DIGITAL TERRESTRE EN EL ECUADOR*. 2010.

[6] TELEVISIÓN DE ALTA DEFINICIÓN. *http://wikitel.info/wiki/Televisi%C3%B3n_de_alta_definici%C3%B3n*. Consultado en Enero 2013.

[7] TELEVISIÓN DIGITAL - VENTAJAS Y DESVENTAJAS. *http://observatoriotecedu.uned.ac.cr/index.php/actualidad/soportes/106-television-digital.html?start=2*. Consultado en Enero 2013.

[8] COMUNIDAD GINGA ECUADOR. *http://comunidadgingaec.blogspot.com/search/label/¿Qué es Ginga%3F*. Consultado en Enero del 2013.

[9] TELEVISIÓN DIGITAL. *http://www.fayerwayer.com/2010/10/desarrollan-el-primer-receptor-europeo-de-television-en-movimiento.* Consultado en Enero 2013.

[10] PÁGINA OFICIAL DE ATSC. *http://www.atsc.org/cms/index.php/standards/standards?layout=default*. Consultado en Enero 2013.

[11] PÁGINA OFICIAL DE DVB. *http://www.dvb.org/*. enero 2013.

[12] ATSC, DTMB, DVB-T/DVB-T2 E ISDB-T. *http://es.dtvstatus.net/*. Consultado en Eenero 2013.

[13] Curasma, Ronald Paucar. *Análisis y Modelamiento de las Técnicas de Canal de Retorno e*. 10 de Diciembre de 2010.

[14] Soares, Luiz Fernando Gomes. *Progrmando em NCL 3.0*. 2012.

[15] Ierusalimschy, Roberto. *Programming in Lua*. 2006.

[16] PÁGINA OFICIAL DE COMPOSER NCL. *http://composer.telemidia.puc-rio.br*. Consultado en Enero 2013.

[17] Lekakos, George. *Interactive Digital Television: Technologies and Applications*. 2008.

[18] Filho, Guido Lemos de Souza. *Ginga-J: The Procedural Middleware for the Brazilian Digital TV System*. 2005.

[19] VENTAJAS FRENTE A LA TV ANALÓGICA. *http://www.canariastdt.es/index.php?option=com_content&view=article&id=21&Itemid=15*. Consultado en Enero 2013.

[20] Serrano Regol, Ivan. *Directrices para el Diseño de Interfaces de Televisión Digital Interactiva.* Consultado en Julio 2013.

CAPÍTULO 2

DISEÑO E IMPLEMENTACIÓN DE UN PLUGIN PARA LA GENERACIÓN DE UN LECTOR DE FEED RSS PARA COMPOSER NCL

2.1 RSS [1], [2], [3], [8]

RSS son las siglas de "*Really Site Summary*", es un formato basado en la estructura XML de descripción y distribución de contenidos, es utilizado para notificar al usuario de actualizaciones o cambios en un servidor web.

Los RSS tienen las extensiones .rss o .xml, pero en realidad son simples archivos de texto donde se muestran referencias de contenidos publicados en un formato específico. En el archivo RSS están los datos de las noticias del sitio web como el título, fecha de publicación o la descripción. El programa que lee el RSS está encargado de suministrar el estilo de los datos que se incluyen en el archivo y presentarlos de una manera atractiva al usuario.

A este método de distribuir y representar contenidos se le denomina "sindicación". Su función es presentar contenidos y noticias sin necesidad de acceder al servidor web. Este método está compuesto de un archivo de texto alojado en el servidor que actúa como fuente de noticias, popularmente son conocidos como *feeds* RSS. Las páginas que utilizan estos servicios tienen un ícono como el que se presenta en la Figura 2.1.

Figura 2.1 Ícono del servicio *feed* RSS [2]

2.1.1 HISTORIA DE RSS [4], [5], [6], [7]

En la Tabla 2.1 se resume la evolución que ha tenido RSS.

Versión	Propietario	Ventajas	Status	Recomendaciones
0.90	Netscape		Obsoleto en 1.0.	No usar
0.91	UserLand	Fácil de abandonar la suscripción.	Oficialmente obsoleto por 2.0.	Empleado para la sindicación básica. Ruta de migración fácil a 2.0 si se necesita más flexibilidad.
0.92, 0.93, 0.94	UserLand	Permite uso más delicado de metadatos 0.91	Obsoleto en 2.0.	Usar 2.0
1.0	RSS-DEV Working Group	Basado en RDF[1], es extensible a través de módulos, no es controlado por un solo proveedor	Núcleo estable de desarrollo, módulo activo.	Está destinada para aplicaciones basadas en RDF o si necesita módulos avanzados de RDF
2.0	UserLand	Extensibilidad a través de módulos, ruta de migración fácil de la rama 0.9x	Núcleo estable de desarrollo, módulo activo.	Uso para fines generales.

Tabla 2.1 Historia del *feed* RSS

[1] RDF (*Resource Description Framework*) es una familia de especificaciones del *World Wide Web Consortium* (W3C) originalmente diseñado como un modelo de datos para metadatos.

2.1.2 VENTAJAS DEL RSS

Las páginas web distribuyen a través de los *feeds* RSS sus últimas actualizaciones, las cuales pueden ser de interés del usuario.

La decisión está del lado del usuario puesto que él es quien elige a qué páginas web suscribirse y cuándo cancelar la suscripción a un *feed* RSS.

El *feed* RSS ahorra tiempo de navegación, en el lector RSS el usuario tendrá un resumen de los artículos para poder decidir qué información quiere obtener.

El *feed* RSS está libre de SPAM[2], esto no ocurre con suscripciones por correo electrónico, en las que además de recibir noticias, existe la posibilidad de recibir SPAM u otra información no deseada. Cuando se suscribe a un *feed* RSS, no se recibirá otra información que la publicada en la misma.

La cancelación de la suscripción a un *feed* RSS es rápida y sencilla, a diferencia del correo electrónico en que el suscriptor tiene que especificar las razones por las que abandona el servicio.

Los *feeds* RSS se reciben de forma gratuita al igual que la mayoría de lectores RSS.

2.1.3 DESVENTAJAS DE LOS RSS

Algunos usuarios prefieren recibir publicaciones electrónicas, noticias y actualizaciones por correo electrónico.

Para los administradores de páginas web es complicado llevar estadísticas de lo que está sucediendo en un *feed* RSS. El número de suscriptores, elementos vistos, cancelaciones de suscripciones, etc. son datos importantes que no se reflejan en los estadísticos del servicio.

El mayor problema en cuanto a datos estadísticos se refiere es el número de cancelaciones de suscripciones y sus razones porque los usuarios solo tienen que

[2] SPAM también conocido como correo basura.

borrar el *feed* RSS del lector. Esto impide obtener una retro alimentación que podría mejorar el marketing de una página web.

2.2 ELEMENTOS DE UN FEED RSS

A continuación se van a presentar los elementos que son necesarios para la creación de un *feed* RSS, como ejemplo modelo se tomará el *feed* RSS de la Escuela Politécnica Nacional.

2.2.1 RSS

El elemento rss pertenece al nivel superior de un *feed* RSS. Un *feed* que cumpla con la especificación RSS debe contener un atributo de versión con el valor "2.0", este elemento es necesario y debe contener solamente un canal. En el Código 2.1 se presenta el elemento rss.

```
<rss version="2.0">
```

Código 2.1 Elemento rss

2.2.2 CHANNEL

El elemento canal o *channel* describe al *feed* RSS, proporcionando información como el título y la descripción, además contiene elementos que representan cambios discretos para el contenido web representado por el *feed*.

Este elemento es obligatorio y debe contener tres elementos secundarios: *description*, *link* y *title*. El canal puede contener los siguientes elementos opcionales: *category, cloud*[3]*, rating*[4]*, copyright*[5]*, docs*[6]*, generator*[7]*, image*[8]*, language*[9]*, ttl*[10]*,*

[3] *cloud* permite que los cambios que se registren en una nube se notifiquen en el *feed*.
[4] *rating* define la clasificación del *feed.*
[5] *copyright* aviso de derechos de autor para el contenido en el canal.
[6] docs especifica un URL con la documentación del formato utilizado en el *feed.*
[7] *generator* especifica el programa que se utilizó para generar el *feed.*
[8] *image* es una imagen que se presenta cuando se abre con un agregador el *feed.*
[9] *language* especifica el lenguaje en el que está escrito el *feed*.
[10] ttl especifica el número de minutos que el *feed* puede estar almacenado en caché antes de actualizar de la fuente.

managingEditor[11], *pubDate*[12], , *skipDays*[13], *lastBuildDate*[14], *skipHours*[15], *textInput*[16] y *webmaster*[17], estos elementos no deberán estar presentes más de una vez, con la excepción de *category* y además puede contener cero o más elementos *item*. El orden de los elementos dentro del canal no es significativo.

2.2.2.1 Title

El elemento título o *title* es el nombre del canal, así es como los usuarios identifican al *feed* RSS. Si tiene un sitio web HTML[18] que contenga la misma información que el archivo RSS el título del canal debe ser el mismo que el título de la página web. El uso de *title* es obligatorio. En el Código 2.2 se presenta el elemento *title.*

```
<title>RSS EPN</title>
```

Código 2.2 Elemento *title* (1)

2.2.2.2 Link

El elemento enlace o *link* identifica el recurso URL[19] del sitio web asociado con el *feed*. El uso de *link* es obligatorio. En el Código 2.3 se presenta el elemento *link.*

```
<link>http://www.epn.edu.ec</link>
```

Código 2.3 Elemento *link* (1)

2.2.2.3 Description

El elemento descripción o *description* contiene datos de caracteres que proporcionan una identificación legible o resumen del *feed*. El uso de *description* es obligatorio. En el Código 2.4 se presenta el elemento *description.*

---

[11] *managingEditor* define la dirección de correo electrónico del editor de contenido del *feed.*
[12] *pubDate* define la última fecha de publicación de los contenidos del *feed.*
[13] *skipDays* especifican los días en los agregadores deben omitir la actualización del *feed*.
[14] *lastBuildDate* define la última fecha de la modificación del *feed.*
[15] *skipHours* especifican las horas en los agregadores deben omitir la actualización del *feed*.
[16] *textInput* especifica un campo de entrada de texto que se debe mostrar con el *feed.*
[17] *webmaster* define la dirección de correo electrónico del *webmaster.*
[18] HTML (*HyperText Markup Language*) lenguaje de marcado hipertextual.
[19] URL (*Uniform Resource Locator*) localizador de recursos uniforme.

```
<description>Subscribete a RSS EPN</description>
```

Código 2.4 Elmento *description* (1)

2.2.2.4 Category

El elemento categoría o *category* identifica una categoría o etiqueta a la que corresponde el *feed*. El uso de *category* es opcional. En el Código 2.5 se presenta el elemento *category.*

```
<category>Próximos eventos</category>
```

Código 2.5 Elemento *category* (1)

2.2.3 ITEM

El elemento *item* representa el contenido publicado en el *feed* como un artículo, entrada de blog, o alguna otra forma de notificación. Un canal puede contener cualquier número de *items* o ninguno.

Un *item* puede contener los siguientes elementos secundarios: *author*[20], *category, comments*[21], *description, enclosure*[22], *guid, link, pubDate, source*[23] y *title*. Todos estos elementos son opcionales, pero un *item* debe contener al menos un *title* o un *description*. Los elementos anteriores no deberán estar presentes más de una vez, con la excepción de *category*. En el Código 2.6 se presenta la estructura de *item.*

```
<item>
    <title>14 Feb 2013 08:00 : CONVOCATORIA: ;BECAS TAIWAN 2013;</title>
    <link>http://www.epn.edu.ec/index.php?option=com_jevents</link>
    <guid>http://www.epn.edu.ec/index.php?option=com_jevents</guid>
    <description><![CDATA[
BECA ACADEMIA SINICA 2013
BECA TAIWÁN 2013
BECA CHINO - MANDARÍN
PROGRAMA DE BECAS ICDE
]]></description>
    <category>Próximos eventos</category>
    <pubDate>Sun, 10 Mar 2013 20:01:08 +0000</pubDate>
</item>
```

Código 2.6 Elemento *item*

[20] *autor* especifica la dirección de correo electrónico al autor del ítem.
[21] *comments* permite que un objeto este vinculado a los comentarios sobre el ítem.
[22] *enclosure* permite que un archivo multimedia que se incluye en el ítem.
[23] *source* especifica un origen de terceros para el ítem.

2.2.3.1 Title

El elemento título o *title* contiene datos que proporcionan el título del *item*. Este elemento es opcional si el *item* contiene un elemento *description*. En el Código 2.7 se presenta el elemento *title.*

```
<title>14 Feb 2013 08:00 : CONVOCATORIA: ;BECAS TAIWAN 2013;</title>
```

Código 2.7 Elemento *title* (2)

2.2.3.2 Link

Elemento enlace o *link* identifica el recurso URL de una página web asociada a un *item*, este elemento es opcional. En el Código 2.8 se presenta el elemento *link.*

```
<link>http://www.epn.edu.ec/index.php?option=com_jevents</link>
```

Código 2.8 Elemento *link* (2)

2.2.3.3 Guid

El elemento guid (*Globally Unique Identifier*) proporciona una cadena que identifica de forma exclusiva al *item*, este elemento es opcional. En el Código 2.9 se presenta el elemento guid.

```
<guid>http://www.epn.edu.ec/index.php?option=com_jevents</guid>
```

Código 2.9 Elemento guid

2.2.3.4 Description

El elemento descripción o *description* tiene el contenido completo del *item* o un resumen de su contenido, una decisión enteramente a discreción del editor. Este elemento es opcional si el *item* contiene un elemento *title*. La descripción debe ser adecuada para su presentación como HTML porque este formato debe codificar como datos de caracteres o bien mediante el empleo de las entidades HTML o una sección CDATA[24]. A diferencia del *description* anteriormente mencionado en este se

[24] CDATA son los datos de los caracteres.

presenta la información de cada noticia, en el otro se presenta la información en general del *feed*. En el Código 2.10 se presenta el elemento *description.*

```
    <description><![CDATA[
BECA ACADEMIA SINICA 2013
BECA TAIWÁN 2013
BECA CHINO - MANDARÍN
PROGRAMA DE BECAS ICDE
]]></description>
```

Código 2.10 Elemento *description* (2)

2.2.3.5 Category

El elemento categoría o *category* identifica una categoría o etiqueta a la que pertenece el *item*. Un *item* puede contener más de un elemento *category*, este elemento es opcional. En el Código 2.11 se presenta el elemento *category.*

```
<category>Próximos eventos</category>
```

Código 2.11 Elemento *category* (2)

2.2.3.6 Pubdate

El elemento pubDate indica la fecha y hora de publicación del *item*, este elemento es opcional. En el Código 2.12 se presenta el elemento pubDate.

```
<pubDate>Sun, 10 Mar 2013 20:01:08 +0000</pubDate>
```

Código 2.12 Elemento pubDate

2.3 METODOLOGÍA DE DESARROLLO DE SOFTWARE [9]

Las metodologías de desarrollo de software nacen ante la necesidad de utilizar una serie de procedimientos, técnicas, herramientas y soporte documental en el momento de desarrollar un producto de software.

Dichas metodologías intentan guiar a las personas que desarrollan software, pero los requisitos de un software a otro son tan variados y cambiantes que han dado lugar a que exista una variedad de metodologías para la creación de software.

2.3.1 METODOLOGÍA ÁGIL [10], [11]

Las Metodologías ágiles o ligeras constituyen un nuevo enfoque en el desarrollo de software, tiene mayor aceptación por parte de los desarrolladores de *e-projects* que las metodologías convencionales como son ISO-9000[25], CMM[26], etc. debido a la simplicidad de sus reglas y prácticas, su orientación a equipos de desarrollo pequeños, su flexibilidad ante los cambios y su ideología de colaboración. En la Tabla 2.2 se pueden observar las diferencias entre un método tradicional con el método ágil.

Las metodologías ágiles son orientadas a la interacción con el cliente y el desarrollador del software, mostrando versiones parcialmente funcionales del software al cliente en intervalos cortos de tiempo, para que pueda evaluar y sugerir cambios en el producto durante el desarrollo.

Los métodos ágiles son adaptables en lugar de predictivos e intentan planear una buena parte del proceso del software, esto funciona bien hasta que los objetos cambian, sin embargo estos son bienvenidos.

A continuación se enumeran algunas de las principales metodologías ágiles:

- Crystal *Methodologies*
- Kanban
- SCRUM
- Lean *Programming*
- Extreme *Programming*

---

[25] ISO-9000 especifica la manera en que una organización opera sus estándares de calidad, tiempos de entrega y niveles de servicio.
[26] CMM (*Capability Maturity Model*) es un modelo de procesos para el desarrollo y mantenimiento de sistemas de software.

- Proceso unificado ágil (UAP)

	Metodología Ágil	**Metodología no Ágil (Tradicional)**
Artefactos	Pocos	Muchos
Roles	Pocos	Muchos
Tipo de contrato	bastante flexible	Existe un contrato prefijado
Cliente	Parte del equipo de desarrollo	Interactúa con el equipo
Grupo de personas	Pequeños (< 10 integrantes)	Grandes
Arquitectura	Menos énfasis	Es esencial
Cambios	Preparado para cambios durante el proyecto	Resistencia ante los cambios
Basado	En la heurística proveniente de producción de código	En normas provenientes de estándares seguidos por el entorno de desarrollo

Tabla 2.2 Comparación entre métodos ágiles y no ágiles

De estos métodos se escogió Kanban para el desarrollo de las aplicaciones.

2.3.1.1 Kanban [12], [13], [14]

2.3.1.1.1 Descripción

Kanban es una palabra escrita en kanji[27], donde kan significa "visual", y ban significa "tarjeta" o "tablero". Es un concepto de producción JIT[28]. Kanban es una tarjeta física

[27] Kanji son los signos más utilizados en la escritura de la lengua japonesa.
[28] JIT (*Just in Time*) es un sistema de organización de la producción para las fábricas, de origen japonés.

que se utiliza en el sistema de producción de Toyota para soportar un control productivo descentralizado por demanda.

Kanban es una herramienta que proviene de la filosofía Lean[29], de tipo "*pull*", lo que significa que los recursos deciden cuándo y cuánto trabajo se comprometen a hacer. Los recursos toman el trabajo cuando están listos en lugar de tener que recibirlo desde el exterior. Kanban se basa en la optimización de procesos continuos y empíricos que corresponden con el principio de Kaizen[30] Lean, enfatiza la respuesta al cambio para seguir un plan, generalmente Kanban permite una respuesta más rápida que los otros métodos ágiles.

2.3.1.1.2 Principios básicos

Kanban presenta los siguientes elementos:

Stream: Permite visualizar el flujo real de trabajo a través de los equipos y brinda información de lo que están realizando en cada ítem. El tablero Kanban está constituido de varias columnas que representan los períodos de desarrollo.

Contenido: Simboliza los tipos de trabajo y asegura de que todo el trabajo este visible para todas las personas que intervienen, esto se realiza con las tarjetas que se colocan en las columnas del tablero Kanban.

Límites: Define claramente los límites de la cantidad de trabajo que el equipo puede soportar dentro de un *stream*. Estos límites indican el punto a partir del cual un equipo no logrará más progresos. Estos límites son definidos con el número de tarjetas máximas en cada columna.

Políticas: Definen documentación comprobables que informan de manera clara y específica lo qué significa prosperar en una tarea o tarjeta de una columna a la siguiente. Las políticas permiten crear confianza en la visualización de todos los interesados.

---

[29] *Lean* es una translación de los principios y prácticas de la manufactura esbelta hacia el dominio del desarrollo de software.
[30] Kaizen es una palabra japonesa que significa "cambio a mejor".

Los principios más importantes de esta filosofía de trabajo son:

- Eliminar los desperdicios en el sistema e incrementar la calidad.
- Determinar el valor del flujo de trabajo[31].
- Remover las demoras del flujo de trabajo aumentando la velocidad al acortar el *cycle time*[32].
- Controlar el WIP[33], dado que el exceso incrementa tanto el riesgo como los desperdicios.
- Evitar el múltiple trabajo de los recursos.
- Visualización del flujo de trabajo a través del tablero Kanban.

2.3.1.1.3 Proceso de desarrollo

Dividir el trabajo en piezas, y escribir cada una de ellas en tarjetas que se colocaran en el tablero Kanban.

Utilizar columnas con nombres para ilustrar dónde se encuentra cada ítem o tarea dentro del flujo de trabajo.

Limitar el WIP, asignando límites específicos de cuántos ítems pueden estar siendo procesados a la vez dentro de cada columna del flujo de trabajo.

Medir el *lead time* también llamado *cycle time* que es el tiempo promedio para completar un ítem, es decir que el ítem haya pasado por todas las columnas del flujo de trabajo hasta llegar al final. Kanban tiene por objetivo optimizar el proceso para hacer el *cycle time* lo más pequeño y predecible que se pueda. En la Figura 2.2 se observa el flujo de trabajo y el WIP en un tablero Kanban.

---

[31] Flujo de trabajo es el estudio de los aspectos operacionales de una actividad de trabajo: cómo se estructuran las tareas, cómo se realizan, cuál es su orden correlativo, cómo se sincronizan entre otras.
[32] *cycle time* tiempo medio para terminar un elemento.
[33] WIP *(Work In Process)* trabajo en curso es el número máximo de tareas que se pueden realizar en cada fase.

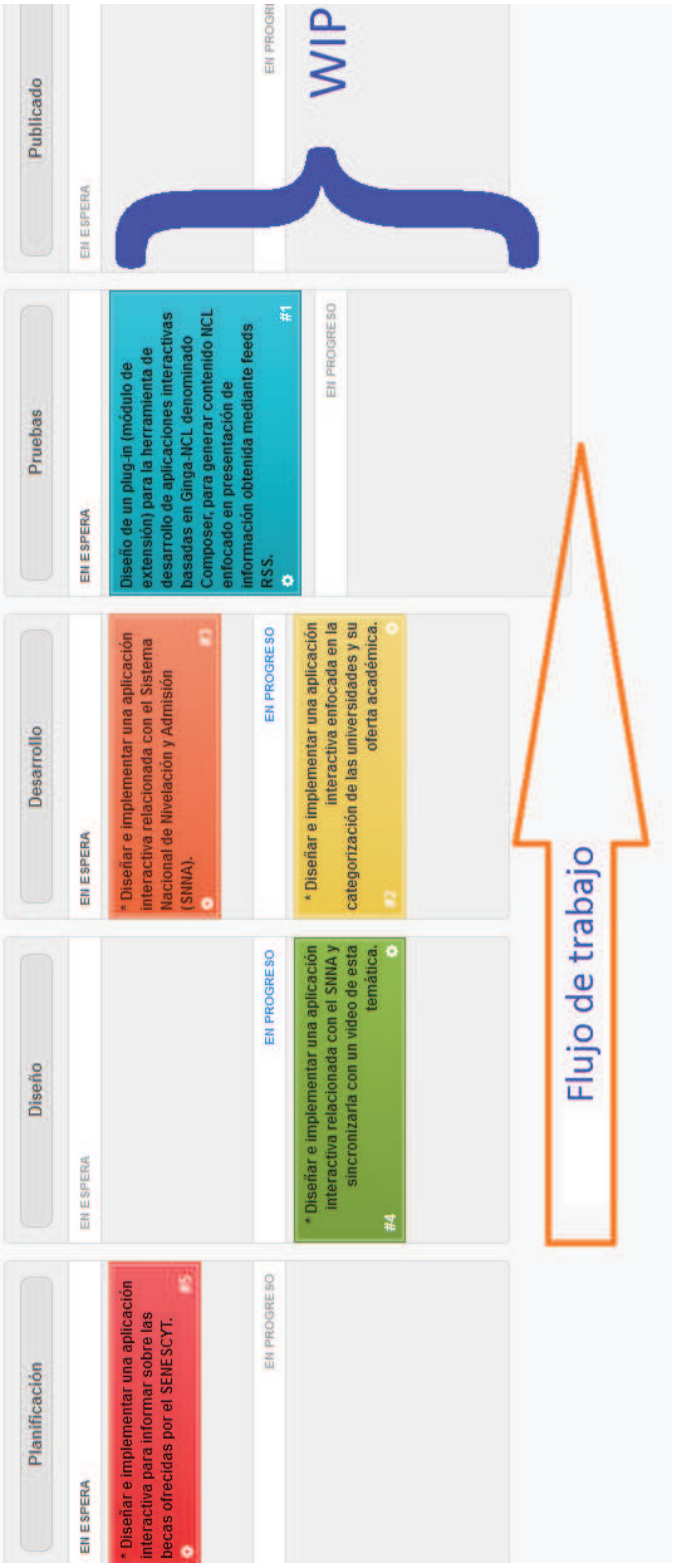

Figura 2.2 Tablero Kanban

2.4 DESARROLLO DEL PLUGIN PARA COMPOSER NCL

Plugin es un módulo de software que añade un servicio específico a un programa. A este software lo ejecuta una aplicación principal, la cual interactúa médiate el API que tienen en común tanto la aplicación principal como dicho *plugin*. Los beneficios que se obtienen son que los desarrolladores de software tanto externos como internos colaboran con la aplicación principal para extender sus funciones y así poder cubrir necesidades específicas. Sin los *plugins* el programa principal puede seguir funcionando sin ningún problema, cuando se activan los *plugins,* el programa principal adquiere las funcionalidades que estos ofrecen para extender las características que tenían en un inicio.

2.4.1 PLUGIN FEED RSS PARA COMPOSER NCL

Al *plugin feed* RSS se lo denominó consumidorRSS, el cual provee una interfaz gráfica para crear los elementos requeridos para visualizar la información obtenida desde un *feed* RSS.

2.4.1.1 Objetivos del consumidorRSS

Para el diseño del *plugin* se tomaron en cuenta los siguientes objetivos:

- Crear un programa que sea idóneo para mostrar la información de un *feed* RSS requerido por un usuario.
- Crear una interfaz gráfica con todos los parámetros definidos en el consumidorRSS, esta interfaz debe ser lo más simple posible ya que no todas las personas que utilizarán el consumidorRSS son programadores. La interfaz también debe ser intuitiva, y solo se debe poner los parámetros estrictamente necesarios para la generación exitosa de aplicaciones interactivas NCL.
- Evitar que el usuario cree incorrectamente la aplicación interactiva porque como se mencionó anteriormente el consumidorRSS está enfocado para personas que no tienen conocimientos de los lenguajes NCL y Lua.

- Integrar el consumidorRSS con *Composer* NCL y que interactúe con los otros *plugins*.

2.4.1.2 Kanban en el desarrollo del plugin para Composer NCL

Acoplando la metodología de Kanban para el desarrollo del consumidorRSS se realizó lo siguiente:

Se dividió en fases de trabajo el desarrollo del consumidorRSS las cuales fueron:

- Conexión con el servidor
- Procesamiento del formato XML
- Presentación de la información seleccionada
- Integración del consumidorRSS a *Composer* NCL
- Diseño de la interfaz gráfica de usuario del consumidorRSS

Estas fases de trabajo son las tarjetas que se utilizaron en el tablero de Kanban ya que cada una de estas fases tenía igual tiempo de desarrollo y de complejidad.

Ya definidas las tarjetas, se colocaron nombres a las columnas para representarlas de una manera adecuada, los nombres que se utilizaron para el tablero Kanban fueron:

- **Planificación:** En esta columna se planificó los alcances y tiempos para que cada tarjeta pueda pasar de una columna a otra.
- **Diseño:** En esta columna se realizó la obtención de datos que estaban relacionados con cada una de las tarjetas de Kanban.
- **Desarrollo:** En esta columna se desarrolló la parte de programación de las tarjetas de Kanban después de tener planificado los alcances y los tiempos; Además se diseñado la información que se desea implementar.
- **Pruebas:** En esta columna se realizó las pruebas de cada tarjeta para observar si se cumplían con las especificaciones planteadas.

- **Publicado:** Esta columna es el final del flujo de trabajo, las tarjetas que pasen por todos las columnas y terminen en esta, tienen el producto listo para la presentación.

Para el WIP cada tarjeta debe ser concluida para que la siguiente entre en proceso activo o de desarrollo ya que todas las tarjetas están relacionadas entre sí. En la Figura 2.3 se presenta el tablero Kanban con todos los elementos que se mencionaron anteriormente.

Figura 2.3 Tablero Kanban del consumidorRSS

2.4.1.3 Aplicación consumidorRSS

El consumidorRSS consta de una programación en C++ con Qt y otra parte en Lua.

2.4.1.3.1 Qt [19], [20]

Qt es una biblioteca multiplataforma[34] que es utilizada para el desarrollo de aplicaciones con interfaz gráfica de usuario (GUI) o sin ésta, como es el caso de aplicaciones basadas en consola.

En 1991, dos programadores noruegos Eirik Eng y Haavard Nord de Quasar *Technologies* crearon un *framework* en C++ que permitiera usar el mismo código en el desarrollo de una interfaz gráfica tanto para Windows como para UNIX. Entre los años 1995 y 1998 se desarrolló con este *framework* un escritorio para GNU/Linux,

[34] Multiplataforma es un atributo de los programas informáticos, que se pueden ejecutar e interpretar en múltiples sistemas operativos.

llamado KDE[35] que hoy sigue siendo uno de los escritorios más usados de Linux. Entre los años 2000 y 2005 se realizó la liberación bajo licencia GPL de las versiones de Qt para los principales sistemas operativos. En 2008, Nokia aseguró el desarrollo de aplicaciones con la adquisición de Qt para sus celulares tanto Symbian[36] como Maemo[37].

Qt es desarrollada como código abierto a través de Qt Proyect, está distribuida bajo los términos de la licencia GNU *Lesser General Public License y* GNU *General Public License* que procura garantizar la libertad de compartir y modificar el software cubierto por ellas.

Qt utiliza el lenguaje de programación C++ de forma nativa, adicionalmente puede ser utilizado con otros lenguajes de programación a través de *binding*[38]. Qt funciona con las principales plataformas de desarrollo. El API cuenta con varios métodos entre ellos: conexión para bases de datos, soporte de red, así como el uso de XML, entre otros.

2.4.1.3.2 ConsumidorRSS con Qt

En esta sección se indicará el código que se realizó en C++ con Qt que se emplea en el consumidorRSS.

Lo que se ha desarrollado es una aplicación que tiene almacenado los archivos Lua, luego esta aplicación los abre, modifica su contenido según lo que el usuario prevea y por último guarda todos los archivos en una carpeta determinada, a continuación se presenta lo realizado.

---

[35] KDE es una comunidad internacional que desarrolla software libre. Normalmente se desarrolla en Qt y C++.
[36] Symbian es un sistema operativo de Nokia.
[37] Maemo proyecto que al final se ha fusionado con Moblin de Intel, para crear MeeGo, y competir con Android de Google.
[38] *binding* es una adaptación de una biblioteca para ser usada en un lenguaje de programación distinto de aquél en el que ha sido escrita.

Para manipular los archivos Lua en el proyecto se ha empleado una carpeta `lua`, que pertenece a `Resources`[39], en la cual se almacenarán estos archivos, esta carpeta constan de un archivo principal llamado `main.lua`, el cual utiliza varias clases que se presentan a continuación con sus archivos que los contienen: `Entities2AccentedChars.lua`, `handler.lua`, `tcp.lua` y `xml.lua`. En la Figura 2.4, se observan los archivos con código Lua que emplea consumidorRSS.

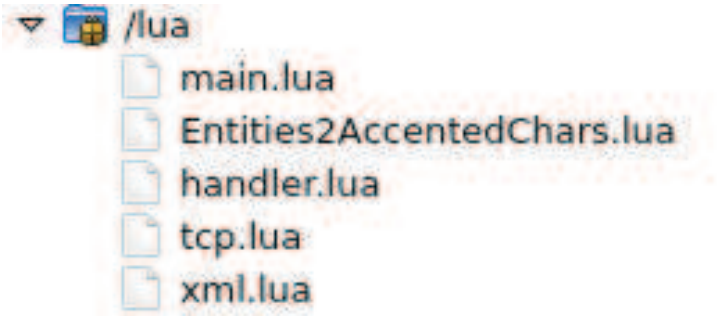

Figura 2.4 Archivos con código Lua para el consumidorRSS

Los archivos guardados en la carpeta `lua` se los deben abrir mediante variables, se pueden leer estos archivos y posteriormente ser usados dentro de la aplicación. Por ejemplo en el Código 2.13 se presenta el código empleado para abrir el archivo `main.lua` y almacenar su información en una variable, para esto se realizó el siguiente proceso: en una variable llamada `file` del tipo `QFile` se especifica la ubicación del archivo `main.lua`.

A través de `QTextStream` que es un flujo que permite contener a `main.lua` en una variable llamada `in` que contiene un puntero de `file`. Con `QString line` se puede leer todo el contenido mediante el parámetro `readAll()` del flujo `in` , por último se utiliza `file.close`, el cual cierra el archivo `main.lua`.

La información ingresada al consumidorRSS por un usuario debe ser validada, si la información es mal ingresada, no será aceptada por el consumidorRSS, el cual

[39] *Resources* en Qt es un mecanismo independiente de la plataforma para almacenar archivos binarios en el ejecutable de la aplicación. Esto es útil si la aplicación necesita siempre un cierto conjunto de archivos y no quiere correr el riesgo de perder los archivos.

enviará un mensaje de advertencia indicando que la información registrada esta errónea. En el archivo `main.lua` se ha especificado un conjunto de parámetros que serán remplazados por datos ingresados por el usuario, estos valores para poder encontrarlos de manera sencilla se los implementó utilizando el símbolo "@" debido a que no existirá este símbolo en el código, por ejemplo `@nombre` remplazará al nombre del archivo `main.lua`.

```
QFile file(":/lua/main.lua");
QTextStream in(&file);
QString line = in.readAll();
file.close();
```

Código 2.13 Código para abrir `main.lua` y guardarlo en una variable

En el Código 2.14 se presenta el código utilizado para remplazar, en el archivo `main.lua`, la información de la fuente de texto ingresado por un usuario. Como se mencionó anteriormente la variable `line` en el Código 2.13 contiene la información del archivo `main.lua`, dentro de este archivo se encuentra `@fuente` el cual será remplazado por la información ingresada por un usuario, contenida en la variable `fuente` esto se lo realiza con la función `replace` la cual acepta dos argumentos que son: la cadena que se desea remplazar y la cadena que lo va a remplazar.

```
line=line.replace(QString("@fuente"),fuente);
```

Código 2.14 Código para remplazar la `fuente` en `main.lua`

Después que el archivo `main.lua` es remplazado con la información seleccionada por el usuario, se deben guardar todos los archivos de Lua en la ruta donde se encuentra la aplicación NCL que fue generada por *Composer* NCL, para utilizarlos dentro de la aplicación desarrollada.

En el Código 2.15 se presenta el código que se utilizó para guardar el archivo `main.lua` en una ruta específica. En primer lugar se crea un objeto `QFile` con el nombre `file2`, que contiene la dirección del lugar donde se va a guardar el archivo, en este caso contiene la ruta donde se encuentra la aplicación NCL guardada en la

variable `ruta` y el nombre que escogerá el usuario para el archivo Lua. Después se crea un objeto `QTextStream` con el nombre de `stream`, que contiene un puntero de `file2`. La información contenida en `line` se la pasa a `stream` utilizando el operador `<<`. Por último se utiliza `file.close`, el cual cierra el archivo. Con este código ya se obtiene el archivo `main.lua` en la ruta donde se encuentra la aplicación NCL con todos los cambios escogidos por el usuario.

```
QFile file2(ruta+"/"+nombrelua+".lua");
QTextStream stream(&file2);
stream<<line;
stream.flush();
file2.close();
```

Código 2.15 Código para guardar `main.lua` dentro de la aplicación NCL

ConsumidorRSS debe generar los códigos NCL dentro de *Composer* NCL, para ello se utiliza el núcleo de *Composer* NCL el cual está encargado de crear los elementos necesarios y la comunicación entre *plugins.*

2.4.2 CONEXIÓN CON EL SERVIDOR

Para poder conectarse con el servidor existen clases externas como son: `LuaSql`, `LuaSocket` y `tcp`. Las dos primeras no forman parte del estándar de GINGA-NCL [24], por lo tanto estas clases no están soportadas por el STB, por esta razón se escogió la clase `tcp`.

2.4.2.1 Clase tcp [17], [15]

La clase `tcp` definida en `tcp.lua` permite crear una conexión TCP entre el *middleware* Ginga y los servicios alojados en un servidor remoto. Las funciones empleadas en la clase `tcp` se encuentran en la Tabla 2.3.

La operación de esta clase se basa en el protocolo TCP, el cual se compone de tres etapas: primero el establecimiento de la conexión, el cual se realiza mediante la función `connect(host,port)`; a continuación se puede realizar la transferencia de

datos, para lo cual se emplea las funciones `send()` y `receive()`; y finalmente se debe cerrar la conexión mediante `disconnect()`. Estas etapas se observan en la Figura 2.5.

Función	Descripción
`connect (host, port)`	Conecta con un servidor determinado mediante TCP `host` = es la dirección del servidor remoto `port` = es el puerto que va a escuchar la conexión
`disconnect ()`	Finaliza la conexión TCP.
`execute (f, ...)`	Es una función que se debe agregar por defecto, llama a todas las funciones de tcp.lua.
`handler (evt)`	Función traductora de eventos[40].
`receive (pattern)`	Recibe respuesta de una solicitud enviada al servidor
`send (value)`	Envía un paquete TCP al host que está conectado

Tabla 2.3 Funciones de la clase `tcp`

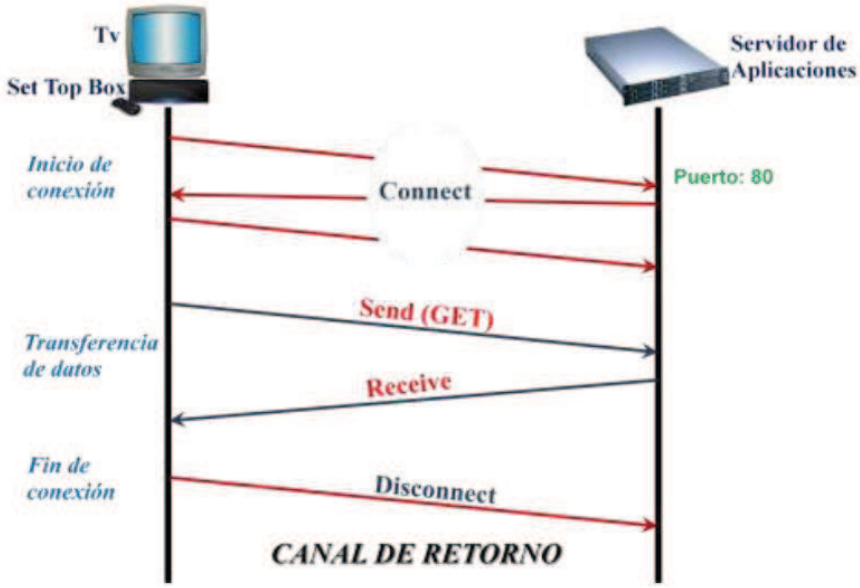

Figura 2.5 Funcionamiento de la aplicación [17]

[40] Traductora de eventos: Se utiliza para controlar los eventos generados por las llamadas a funciones en la clase `tcp`.

En el Código 2.16 se puede observar parte del código utilizado en `main.lua`, el cual presenta la función `tcp.execute` de la clase `tcp`, e incluye las modificaciones necesarias para realizar una conexión hacia un servidor remoto.

```
tcp.execute(
      function ()
         local host = " www.epn.edu.ec"
         local path = "/index.php?option=com_jevents&task=modlatest.rss&
format=feed&type=rss&modid=145"
         tcp.connect(host, 80)
         local url = "http://"..host..path
         local request = "GET "..url.." HTTP/1.0\n"
         request = request .. "Host: "..host.."\n\n"
         tcp.send(request)
         local feedEPN = tcp.receive("*a")
         tcp.disconnect()
      end
   )
```

Código 2.16 Función `tcp.execute` definida en `main.lua`

La función `tcp.execute` realiza las tres etapas de TCP. Mediante la función `tcp.conect(host,80)` se establece la conexión con el servidor remoto mediante el protocolo TCP, el servidor estará determinado por el par conformado por el nombre de equipo y su puerto, en este caso el nombre del servidor es “www.epn.edu.ec” y el puerto 80.

Con la variable `request` se genera un paquete de datos empleando el protocolo HTTP, el cual contendrá el comando que debe ejecutarse en el sitio remoto, así como el URL del recurso que se solicita. En este caso, el paquete de datos define el comando `GET` con el cual se solicita el recurso especificado en la variable `url` y además se indica la versión del protocolo HTTP/1.0.

Mediante `tcp.send(request)` se envía la solicitud HTTP al servidor remoto. El servidor responderá con el recurso solicitado, para obtener dicho recurso se emplea el método `tcp.receive("*a")`, el parámetro `*a` permite obtener todos los datos sin tener que realizar llamadas sucesivas a `tcp.receive()`. La variable `feedEPN` contendrá la información del *feed* RSS en formato XML.

Por último se cierra la conexión TCP mediante `tcp.desconnect()`. Con esta clase se realizará la conexión hacia el *feed* RSS y se trae toda la información en formato XML, el siguiente paso es seleccionar la información que se requiere para ser mostrada.

2.4.3 TRATAMIENTO DEL FORMATO XML [18]

Una vez que se obtiene la información en formato XML se la debe procesar para su presentación al usuario. Para simplificar las tareas de obtención y presentación de información se dispone de dos clases: `handler` y `xml` que se utilizan para obtener la información específica de las etiquetas XML.

La clase `handler` definida en `handler.lua` ofrece varias funciones: `showTable` que retorna una representación de cadena de la tabla ingresada, `printHandler` imprime información para poder realizar el seguimiento de eventos para la depuración, `simpleTreeHandler` genera una tabla Lua formada por una cadena XML y `domHandler` permite generar una estructura de árbol de nodos con un solo nodo primario llamado *root*. La función más importante es `simpleTreeHandler` que básicamente se comporta como un controlador simplificado que intenta generar una estructura basada en una tabla. La estructura de árbol XML se asigna automáticamente a una estructura de tabla recursiva con nombres de nodos como llaves y elementos secundarios, ya sean como tablas de valores o como cadenas de texto.

La clase `xml` especificada en `xml.lua` proporciona una API parcialmente orientada a objetos con funcionalidad de división en componentes. La instancia handler recibe devoluciones de llamada por cada elemento XML procesado. Esto proporciona un

analizador de secuencias XML, sus datos pasan a la instancia del analizador a través del método `parse`.

La clase `Entities2AccentedChars` especificada en `Entities2AccentedChars.lua` solo convierte el código HTML en letras con acento o caracteres especiales del idioma Español. Esta clase cuenta con una tabla en la que se asigna los caracteres y sus códigos con referencia de ISO8859-1[41]. La función más importante de esta clase es `ConvertString()` la cual sustituye los códigos HTML si existiese en letras que ya están en formato ISO8859-1.

En el Código 2.18 se presenta un segmento de código que permite procesar el contenido XML. La instancia `tablafeed` permitirá crear una estructura de árbol a través de la función `simpleTreeHandler`, que a posteriori almacenará la información obtenida desde el árbol y se usará para generar la tabla Lua. La variable `feedEPN` contiene el *feed* RSS, por ejemplo esta variable contendrá el segmento de código presentado en el Código 2.17 si se accede al *feed* RSS de la Escuela Politécnica Nacional. Este segmento de código fue obtenido a través del Código 2.16.

La función `ConvertString()` acepta como argumento `feedEPN` el cual procesará de la siguiente manera, si el código almacenado en `feedEPN` es HTML entonces procede a cambiar las letras con acento o caracteres especiales del idioma, en caso que el código sea XML no realiza ningún cambio en particular, como el Código 2.17 presentado esta en XML no realiza ninguna tarea.

Posteriormente se crea una instancia `xmlParser()` con el nombre de `xmlparser`, es necesario pasar como parámetro `tablafeed`, este parámetro se utiliza para convertir la cadena XML a otro tipo de formato especificado en la clase `handler`, en este caso es un árbol Lua. El método `parse` de la instancia del analizador XML (`xmlparser`) acepta como argumento la variable donde contiene la información del

[41] ISO8859-1 es una norma de la ISO que define la codificación del alfabeto latino, incluyendo los diacríticos y letras especiales.

feed RSS a ser analizada, en este caso es `feedEPN`. Finalmente llevado a cabo este proceso, la instancia `tablafeed` contiene la tabla Lua, necesaria para ser utilizada en el programa NCL y que será presentado en pantalla.

```
<rss version=”2.0” xmlns:atom=”http://www.w3.org/2005/Atom">
        <channel>
                <title>RSS EPN</title>
                <description>Subscribete a RSS EPN</description>
                <link>http://www.epn.edu.ec</link>
                <lastBuildDate>Thu, 11 Jul 2013 16:06:06 </lastBuildDate>
                <generator>Joomla OpenSourceContent Management</generator>
        </channel>
</rss>
```

Código 2.17 Segmento de código del *feed* RSS

```
dofile(“Entities2AccentedChars.lua”)
dofile("xml.lua")
dofile("handler.lua")

local tablafeed = simpleTreeHandler()
feedEPN = ConvertString(feedEPN)
local xmlparser = xmlParser(tablafeed)
xmlparser:parse(feedEPN)
tablafeed.root.rss.channel.link
```

Código 2.18 Parte del código para procesar el contenido XML definido en `main.lua`

Para obtener la etiqueta *link* del segmento de código del *feed* RSS presentado en el Código 2.17 se emplea el código `tablafeed.root.rss.channel.link` que se encuentra en el Código 2.18*,* este segmento de código retornará el valor de `http://www.epn.edu.ec`.

Para obtener información de una etiqueta del *feed* RSS se debe agregar la instancia que se utilizó para declarar `simpleTreeHandler()` que en este caso es `tablafeed` el cual contiene todo el árbol XML, luego se coloca `root` y se puede desplazar por todos las etiquetas que se encuentran en el *feed* RSS, no todos son cadenas de texto, también se encuentran tablas de valores.

2.4.4 PRESENTACIÓN DE LA INFORMACIÓN SELECCIONADA

Para poder imprimir la información generada mediante Lua en NCL se necesita un módulo llamado *canvas.* Este módulo permite realizar operaciones gráficas durante una presentación del elemento media, como dibujar texto, líneas, imágenes, etc.

2.4.4.1 Primitivas canvas [16]

Las primitivas son todos los métodos que toman en cuenta los atributos de *canvas.*

En la Figura 2.6 se presentan los elementos que intervienen en *canvas* como son: `height` que es el altura de *canvas*, `width` que es la ancho de *canvas* y las coordenadas de `x` que es el lugar donde va a comenzar *canvas* en el eje "x" y `y` que es el lugar donde va a comenzar *canvas* en el eje "y", estos elementos se utilizan tanto para las primitivas como para los atributos.

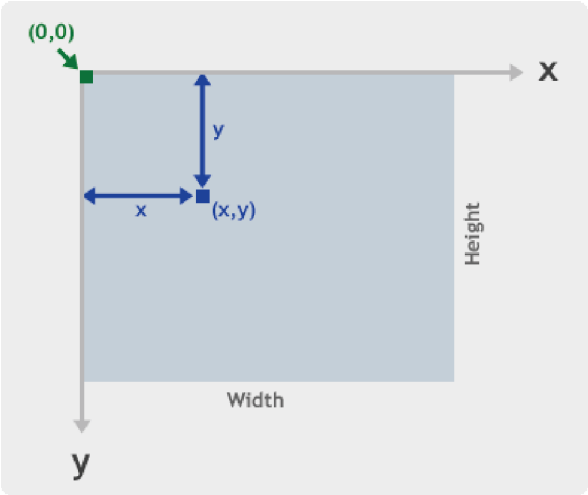

Figura 2.6 Lienzo de *canvas* [17]

El código de la primitiva `canvas:drawRect` se presenta en el Código 2.19, el cual recibe los siguientes parámetros: `mode` que puede recibir las cadenas `fill` para dibujar y rellenarlo o `frame` simplemente para dibujar el rectángulo; los parámetros `x` y `y` que son la posición donde comenzará el rectángulo y los parámetros `width` y `height` que son el ancho y la altura del rectángulo.

canvas:drawRect (mode: "fill" o "frame"; x, y, width, height: número)	
Argumentos	
mode	Modo de dibujo
x	Coordenada x del rectángulo
y	Coordenada y del rectángulo
width	Ancho del rectángulo
heigth	Altura del rectángulo

Código 2.19 Primitiva `canvas:drawRect`

El código de la primitiva `canvas:drawText` se utiliza para introducir el texto previamente seleccionado, esta primitiva se puede emplear usando los atributos como `attrColor` y `attrFont` para asignar el color, tamaño y tipo fuente de letras. En el Código 2.20 se presentan los argumentos de la primitiva `drawText`.

canvas:drawText (text: string; x, y: número)	
Argumentos	
text	Texto que va a ser dibujado
x	Coordenadas de texto en x
y	Coordenadas de texto en y

Código 2.20 Primitiva `canvas:drawText`

La primitiva `canvas:measureText` devuelve las coordenadas de los límites de un texto dado, como si se hubiese dibujado en las coordenadas en `x` y en `y`. En el Código 2.21 se presentan los argumentos de la primitiva y los valores que retorna `measureText`.

canvas:measureText (x, y: numero; text: string) -> x1, x2, y1, y2: número	
Argumentos	
x	Coordenada x del texto
y	Coordenada y de texto
text	Texto a medir
Valores de retorno	
x1	Coordenada del extremo izquierdo en x
y1	Coordenada del extremo izquierdo en y
x2	Coordenada del extremo derecho en x
y2	Coordenada del extremo derecho en y

Código 2.21 Primitiva `canvas:measureText`

La primitiva `canvas:flush` limpia el contenido de *canvas* después de una serie de dibujos y operaciones que se realizaron utilizando *canvas.* En el Código 2.22 se presenta la primitiva `flush`.

canvas:flush ()

Código 2.22 Primitiva `canvas:flush`

2.4.4.2 Atributos de canvas [16]

Todos los atributos tienen el prefijo `attr` y se utilizan para obtener y establecer los atributos de las primitivas con las excepciones que se especifican más adelante.

Cuando se llama a un atributo sin parámetros de entrada se devuelve el valor del atributo actual. Por otro lado, cuando se llama un atributo con los parámetros de entrada estos deben ser usados como los nuevos parámetros de los atributos.

El atributo `canvas:attrSize` devuelve las dimensiones de *canvas.* Es importante tener en cuenta que las dimensiones de *canvas* no se pueden cambiar si ya han sido especificadas. En el presente Código 2.23 se observan los valores que retorna el atributo `attrsize`.

canvas:attrSize () -> *width, height*: número

Valores que retorna

Width	Ancho del *canvas*
Height	Altura del *canvas*

Código 2.23 Atributo `canvas:attrSize`

El atributo `canvas:attrColor` permite definir el color que será usado por las primitivas para dibujar lo especificado. Los argumentos del atributo se presentan en el Código 2.24 y son `RGBA: R` representa el componente de color rojo; `G` representa el componente de color verde; `B` representa el componente de color azul; `A` representa el componente alfa que sirve para transparencia; estos son los colores base para crear cualquier color, sus valores pueden variar desde 0 (transparencia total) hasta 255 (opacidad completa).

canvas:attrColor (R, G, B, A: número)

Argumentos

R	Componente rojo	B	Componente azul
G	Componente verde		
A	Componente Alfa		

Código 2.24 Atributo `canvas:attrColor` (1)

El atributo `canvas:attrColor` también permite definir el color que será usado por las primitivas para dibujar lo especificado. El argumento del atributo se presenta en el Código 2.25 y es `nombre_color`, este argumento tiene 16 colores predefinidos en Lua que son: `white`, `aqua`, `lime`, `yellow`, `red`, `fuchsia`, `purple`, `maroon`, `blue`, `navy`, `teal`, `green`, `olive`, `silver`, `gray` y `black`.

canvas:attrColor (nombre_color: string)

Argumentos

Nombre_color	Nombre del color

Código 2.25 Atributo `canvas:attrColor` (2)

El atributo `canvas:attrFont` permite cambiar la fuente, tamaño y estilo empleado en el texto a presentar en *canvas*. Los argumentos del atributo se presentan en el Código 2.26 y son: `face` es la fuente del texto, las fuentes que están disponibles son: `times`, `courier` y `helvética`; `size` es el tamaño del texto, el tamaño se específica en píxeles, representa la altura que va a tener una línea escrita; `style` es el estilo del texto, los posibles estilos son: `bold`, `italic`, `bold-italic` y `nill`.

canvas:attrFont (face: string; size: número; style: string)	
Arguments	
face	Nombre de la fuente
size	Tamaño del texto
style	Estilo del texto

Código 2.26 Atributo `canvas:attrFont`

Estos son los elementos básicos para manipular *canvas* en un programa Lua y poderlo presentar en una aplicación NCL, la cual debe tener un elemento media con una área determinada y con un descriptor para poder observarlo. Con todos los elementos mencionados se puede realizar un programa básico de una aplicación para televisión digital. Para una mejor explicación a continuación se presenta un ejemplo en el Código 2.27 el cual contiene las variables `tancho` y `taltura` con estas dos variables se obtiene el tamaño de *canvas* mediante `canvas:attrSize()`.

Con el atributo `canvas:attrColor(0,0,0,255)` se escogió el color, en este caso es blanco, por lo que se utilizará la primitiva `canvas:drawRect("fill",0,0,cw,ch)` la cual dibujará un rectángulo sin relleno y transparente con el tamaño total de *canvas.*

En este punto se utiliza nuevamente el atributo `canvas:attrColor("teal")` el cual cambia el color a verde azulado para utilizarlo con el siguiente atributo `canvas:attrFont("times",24)` el cual utiliza como fuente `times` y como tamaño `24` estos dos atributos son utilizados por la primitiva

`canvas:drawText(14, 14, tablafeed.root.rss.channel.link)` el cual imprimirá lo que contenga la variable `tablafeed.root.rss.channel.link` en el *canvas* de acuerdo a los valores que se especifican en ancho y altura.

```
local tancho, taltura = canvas:attrSize()
canvas:attrColor(0, 0, 0, 255)
canvas:drawRect("fill", 0, 0, tancho, taltura)
canvas:attrColor("teal")
canvas:attrFont("vera", 24)
canvas:drawText(14, 14, tablafeed.root.rss.channel.link)
```

Código 2.27 Ejemplo de los atributos y primitivas de *canvas*

2.4.5 INTEGRACIÓN DEL CONSUMIDORRSS A COMPOSER NCL [22]

ConsumidorRSS tiene que integrarse con *Composer* NCL dado que va a ser un *plugin* del mismo por lo cual debe utilizarse dos interfaces, escritas en C++, que son: IPluginFactory e IPlugin.

2.4.5.1 IPluginFactory

El IPluginFactory es responsable de crear nuevas instancias de IPlugin, entregando información global sobre el *plugin* creado, tales como su fabricante, la versión, entre otros. Cada vez que se inicia un nuevo documento *Composer* NCL, se trata de crear una nueva instancia del *plugin* llamando al método de IPluginFactory denominado `createPluginInstance()`.

En el Código 2.28 se presenta parte del código que se utiliza en `IPluginFactory` que se utilizó en consumidorRSS. Consta de las variables con la información de dicho *plugin* como son el nombre, versión, quien lo realizó, *copyright*[42], una descripción rápida del *plugin*, la URL que está asociado y la categoría a la que pertenece. Esta información se presenta en la ayuda del *plugin* en *Composer* NCL.

[42] *Copyright* conocido como derechos de autor es un conjunto de normas jurídicas y principios que lo regulan.

```
    QString name() const {return "consumidorRSS";}
    QString version() { return "1.1"; }
    QString compatVersion() {return "1.1";}
    QString vendor() {return "Escuela Politécnica Nacional Laboratorio de TV Digital";}
    QString copyright() {return "Escuela Politécnica Nacional";}
    QString description() {return "RSS Feed plugin is an application that gives us the
files to have a connection with a website that has RSS Feed";}
    QString url() {return "http://ginga.epn.edu.ec";}
    QString category() {return "Feed RSS";}
```

Código 2.28 `IPluginFactory` de consumidorRSS

2.4.5.2 IPlugin

El IPlugin es el responsable de comunicarse directamente con el núcleo de *Composer* NCL y de interactuar con los demás *plugins.* Cuando se crea un documento de Composer NCL se requiere una nueva instancia del *plugin*. El IPlugin está vinculado con la instancia del documento y será llamado cada vez que haya cambios en cualquier *plugin* de *Composer* NCL. De esta forma, el IPlugin también es capaz de exigir cambios en el documento.

Además de estas clases se debe ocupar funciones especiales que tiene Qt, que son *signals* y *slots* para que se comunique el *plugin* que se está desarrollando con el núcleo de *Composer* NCL y con los demás *plugins*. En [22] se encuentra el código para realizar un *plugin.*

En el Código 2.29 se presenta parte del código que se implementó en `IPlugin` para consumidorRSS el cual cuenta con `QWidget* getWidget()`, el núcleo llama a este método para notificar al *plugin* que el usuario solicita guardar este documento. Ciertos *plugins* tienen que guardar una configuración especial para un documento en particular, en el caso que se vuelva a abrir un archivo diferente. `init()` llama al *plugin* para actualizar su modelo interno, esta llamada es invocada por el núcleo en dos situaciones:

- Cuando el usuario solicita una actualización de todos los plugins.
- Cuando el *plugin* se carga en tiempo de ejecución.

En el segundo caso, el *plugin* está cargado, pero no recibirá las modificaciones anteriores, por lo que tiene que ser forzado a cargar el documento. `public slots` se utiliza después de que el *plugin* se conecte con el núcleo. Después del retorno de `init`, el *plugin* es capaz de recibir mensajes del núcleo.

```
QWidget* getWidget();
   void init();
public slots:
```

Código 2.29 Parte del codigo de `IPlugin` de consumidorRSS

2.4.5.3 Signals y Slots [19], [20]

Qt define una forma dinámica de comunicar eventos con los cambios de estado que estos pueden provocar y las reacciones de los mismos denominadas *Signals* y *Slots*. Lo original de este mecanismo es que los objetos que se comunican dinámicamente, no requieren conocerse mutuamente. Para ello QObject la clase base de gran parte de las clases que conforman Qt, incorpora un método llamado “connect()”,que se encarga de establecer dicha comunicación entre los dos objetos. Todo objeto derivado de QObject puede poseer dos tipos de funciones propias especiales: *Signals* y *Slots*.

Signals son funciones que permiten emitir una señal cuando hay algún cambio de estado en el objeto al que pertenecen.

Slots son funciones que marcan el final de la conexión y que ejecutan una serie de acciones una vez que reciben el mensaje de la señal.

En la Figura 2.7 se puede observar que una señal puede conectarse a más de un *slot* para llevar a cabo diferentes tipos de actividades. Igualmente varias señales diferentes pueden conectarse con un mismo *slot* y se tendrá una actividad que puede ser desplegada de varias formas.

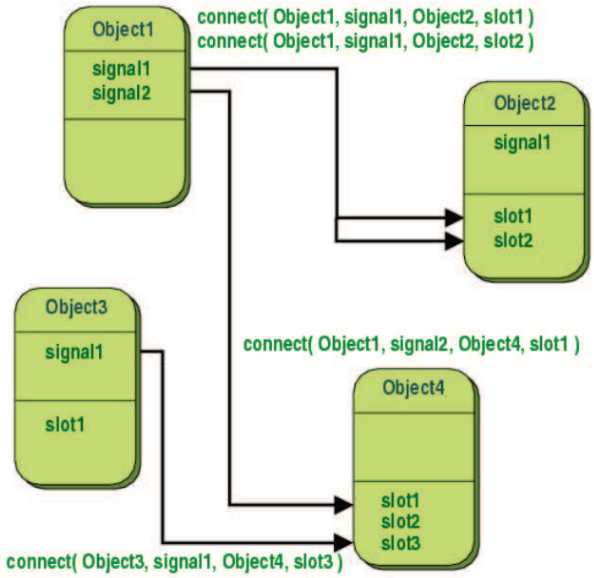

Figura 2.7 Funcionamiento de *signals* y *slots* [19]

Propiedades de la función *Slot:*

- Es implementada como una función ordinaria.
- Puede ser llamada como una función ordinaria.
- Un *Slot* puede ser declarado en la zona *private*, *protected* o *public* de una clase, pero eso solo afectará si es llamada como función y no como *Slot*. Como *Slot* siempre podrá ser conectada con objetos de otra clase independientemente de la zona donde se haya definido en su clase. Se declara en la sección con la palabra *Slots*.
- Como función puede devolver un valor de retorno.
- Cualquier número de *Signals* puede ser conectadas a un mismo *Slot*.

Propiedades de la función *Signal*:

- *Signal* es una función que siempre retorna void.
- *Signal* puede ser conectada a cualquier número de *slots* a la vez.
- Los *Slots* son activados por las *signals* en orden arbitrario.
- *Signal* es emitida desde cualquier parte del código de la clase mediante `emit (emit nombre_signal(parametros))`.

Con todos estos elementos se puede crear un *plugin* integrado con *Composer* NCL. Para la integración con los demás *plugins* se debe utilizar la clase del núcleo que esta implementada en *Composer* NCL.

2.4.5.4 Núcleo Composer NCL [23]

Para la integración del *plugin* con *Composer* NCL y con los demás *plugins*, se debe utilizar el módulo "Composer-core". Este módulo contiene clases y funciones que ayudan con la integración del *plugin*. Para integrar el *plugin* se deben utilizar *Signals* y *Slots*, *Signals* se utiliza para la comunicación con los otros *plugins* y *Slots* se utiliza para la comunicación con el núcleo de *Composer* NCL.

2.4.5.4.1 Slots del núcleo de Composer NCL

Los *Slots* son llamados por el núcleo cuando se realiza una acción específica como se detalla a continuación:

`composer::extension::IPlugin::onEntityAdded` es llamado cuando se agrega una entidad, este *slot* requiere que se pase como argumento el ID del evento producido y un puntero a la entidad.

`composer::extension::IPlugin::onEntityChanged` es llamado cuando se cambia una entidad, este *slot* requiere que se pase como argumento el ID del evento producido y un puntero a la entidad.

`composer::extension::IPlugin:: onEntityRemoved` es llamado cuando se elimina una entidad, este *slot* requiere que se pase como argumento el ID del evento producido y un ID del elemento eliminado.

`composer::extension::IPlugin::errorMessage` es llamado cuando se produce un error provocado por una instancia del *plugin*, este *slot* requiere que se pase como argumento el error que se produjo. Todos los *slots* presentados se utilizaron en el consumidorRSS y se los presentan en el Código 2.30.

```
Void onEntityAdded(Qstring ID, Entity *);
void onEntityChanged(QString ID, Entity *);
void onEntityRemoved(QString ID, QString entityID);
void errorMessage(QString error);
```

Código 2.30 *Slots* utilizados en el consumidorRSS

2.4.5.4.2 Signal del núcleo de Composer NCL

Para agregar elementos al documento NCL se debe utilizar la siguiente *signal* `Composer::extension::IPlugin::addEntity()`, la que acepta los siguientes parámetros: `QString type`, con el que se especifica el tipo de elemento NCL que se desea agregar en el documento NCL; `QString parentEntityId`, para especificar el identificador único del elemento padre de la entidad que se desea agregar; `QMap<QString,QString>&atts`, con el que se especifican los parámetros de la entidad y `bool force`, si está en `true` la entidad va a ser agregada incluso si esto causo un error.

En el Código 2.31 se presentan los elementos que intervienen para agregar un elemento NCL en *Composer*, En la primera línea se aprecia lo siguiente a través de `Entity *body` que es un puntero de tipo `Entity` se ha obtenido el primer elemento *body* del documento NCL utilizando `Project->getEntitiesbyType().first`.

La clase `composer::core::model:: Project` es una estructura de datos que mantendrá a todas las entidades y datos del *plugin* en un mismo lugar, con esta clase se utilizó la función `getEntitiesbyType` que tiene como parámetro `QString _type` en el que se ingresa el nombre de la entidad a buscar, la cual despliega una lista con todos los elementos que se encuentra en el archivo NCL y los retorna en forma de `QList<Entity*>`.

La clase `composer::core::model::Entity` es la clase interna principal del núcleo de *Composer* NCL ya que su modelo interno es un árbol de entidades. En la segunda línea se puede observar que se obtiene el ID del *body*, el cual es necesario para agregar la entidad, el puntero `body` utiliza la función `getUniqueId()` para retornar el valor mencionado, toda la información obtenida se guarda en la variable `parentEntityId`.

En la tercera línea se llama a al *signal* para que se agregue el código adecuado y el elemento respectivo. Para utilizar la *signal* `addEntity()` se deben agregar los siguientes parámetros:

`type` que en este caso puede ser *link*, *media*, *port* entre otros que tengan como elemento padre a *body*; a `parentEntityId` se debe añadir el valor del identificador único del elemento *body*; `atts` son las propiedades de los elementos hijos del *body* que se desea agregar y el valor `force` en donde siempre se encuentra `false` para que si existiese un error no se agregué el elemento.

```
1   Entity *body = project->getEntitiesbyType("body").first;
2   parentEntityId = body->getUniqueId();
3   emit addEntity(type, parentEntityId, atts, force);
```

Código 2.31 *Signal* `addEntity()` para agregar un elemento hijo de *body*

2.4.6 DISEÑO DE LA INTERFAZ GRÁFICA DE USUARIO DEL CONSUMIDORRSS

El consumidorRSS se integrará a *Composer* NCL en forma de un *plugin*. La aplicación *Composer* NCL se basa en Qt, por tanto el *plugin* debe ser escrito en el mismo lenguaje, por esta razón se utilizó Qt Creator.

2.4.6.1 Qt Creator [19], [20], [21]

Figura 2.8 Ícono de Qt Creator [19]

Qt Creator es un IDE multiplataforma desarrollado por TrollTech que se ajusta a las necesidades de los desarrolladores de Qt. Qt Creator es parte de Qt Proyect, en la Figura 2.8 se puede observar su ícono. En la Tabla 2.4 se presentan los sistemas operativos que soportan Qt Creator y sus características para su funcionamiento.

Sistemas Operativos	Versión	Requerimientos
GNU/Linux	2.6.x	Qt instalado
Mac OS	10.4 o superiores	Qt instalado
Windows	Windows XP o superiores	MinGW y Qt para MinGW

Tabla 2.4 Requisitos para la instalación de Qt Creator

Qt Creator proporciona características que ayudan a los nuevos usuarios de Qt a aprender y comenzar a desarrollar código rápidamente, también aumenta la productividad de los desarrolladores con experiencia en Qt. Algunas de las características que tiene este *framework* son:

- Editor de código con soporte para C++, QML[43] y ECMAscript[44]
- Herramientas para la navegación rápida en el código
- Resaltado de sintaxis y auto-completado de código
- Control estático de código y estilo a medida que se escribe
- Ayuda sensitiva al contexto

[43] QML (Qt *Meta Language*) es un lenguaje basado en JavaScript creado para diseñar aplicaciones enfocadas a la interfaz de usuario.
[44] ECMAscript define un lenguaje de tipos dinámicos ligeramente inspirado en Java y otros lenguajes del estilo de C.

- Plegado de código
- Paréntesis coincidentes y modos de selección

Qt Creator también posee un depurador visual para C++, es consciente de la estructura de muchas clases de Qt lo que aumenta la capacidad de presentar los datos con claridad. Además Qt Creator muestra la información sin procesar, originaria de GDB[45] de una manera clara y concisa.

Qt Creator se utiliza para la creación y el diseño de formularios para proyectos C++ que permiten diseñar rápidamente widgets y diálogos. Los formularios son funcionales y pueden ser visualizados inmediatamente para asegurarse de que se apreciarán exactamente como se los diseño.

2.4.6.2 Diseño de la interfaz gráfica del consumidorRSS con Qt Creator

Para el diseño de la interfaz gráfica se deben considerar los datos que se van a ocupar para la creación de los elementos Lua. El diseño de esta interfaz debe ser lo más simple posible para cubrir los objetivos que se plantearon al inicio del capítulo. Como el consumidorRSS diseñado está integrado con *Composer* NCL es necesario tomar en cuenta que con un solo botón se generarán todos los códigos deseados con sus validaciones.

Los datos que se pueden utilizar son: “Página del *feed* RSS”, “Fuente del texto”, “Color del texto”, “Tamaño del Texto” y “Nombre del archivo”. Estos datos se utilizan para la configuración de los archivos Lua, los cuales realizan la conexión, obtienen y procesan la información del *feed* RSS.

En “Pagina del *feed* RSS” se debe ingresar el URL del *feed* RSS, con este dato es posible obtener la información que se desea presentar.

En “Fuente del texto” se define la fuente que se empleará para la visualización de los resultados, las posibles fuentes son: `Times`, `Courier`, `Helvetica` y `Vera` las cuales están especificadas en [24].

[45] GDB o GNU *Debugger* es el depurador estándar para el compilador GNU.

En “Color del texto” se define el color del texto para su presentación, pudiendo escoger de entre las siguientes opciones: `white`, `aqua`, `lime`, `yellow`, `red`, `fuchsia`, `purple`, `maroon`, `blue`, `navy`, `teal`, `green`, `olive`, `silver`, `gray` y `black` los cuales están especificadas en [24].

En “Tamaño del texto” se define el tamaño de los datos a ser presentados, el rango de valores permitidos está entre 1 a 40.

En “Nombre del Archivo” se debe ingresar el nombre del archivo principal, por ejemplo: `main.lua`.

Para utilizar esta información se usará como diseño la Figura 2.9 en la cual se presenta el primer *sketch* que se realizó, en esta figura se encuentran todos los parámetros que anteriormente se mencionaron.

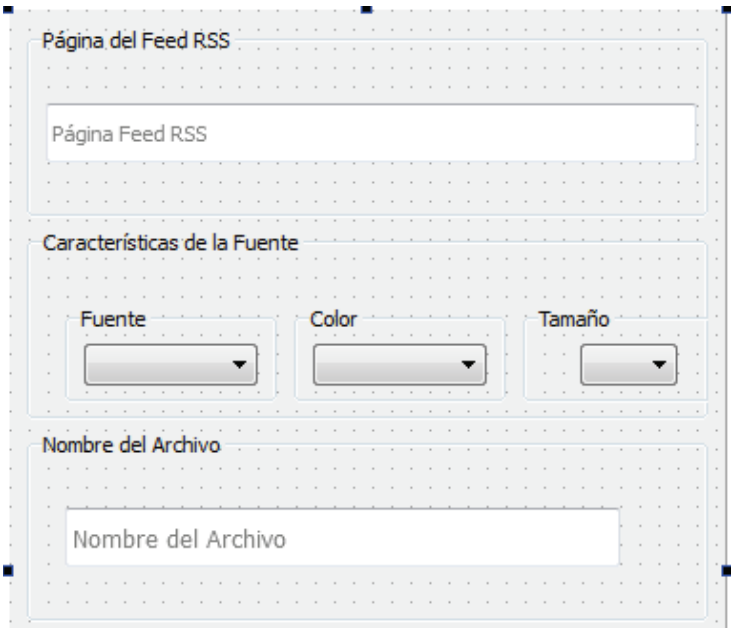

Figura 2.9 Primer *sketch* del *plugin* RSS

La información “Pagina *feed* RSS” se utilizó con un QLineEdit como se observa en la Figura 2.10. Esta información no debe tener espacios en blanco o estar vacía, si los existiese enviará un mensaje de error.

Figura 2.10 QLineEdit para “Página *feed* RSS”

Las informaciones de “Fuente del texto”, “Color del texto y “Tamaño del texto” se agregaron todas en un QGroupBox para agruparlas ya que todas están relacionadas con parámetros que se utilizarán para la presentación del texto de la información del *feed* RSS.

La primera información que se utilizó fue “Fuente de texto”, la misma que se especificó anteriormente, para la presentación de estos elementos se utilizó QComboBox porque el usuario solo podrá escoger las opciones que están soportadas como se observa en la Figura 2.11.

Figura 2.11 QComboBox “Fuente de texto”

La segunda información que se ocupó fue “Color de texto”, detalladas anteriormente, para la presentación de estos elementos se utilizó QComboBox como se observa en la Figura 2.12, en la cual se aprecia que además del nombre del color existe un cuadro con el color que lo representa.

La tercera información utilizada fue “Tamaño del texto”, los valores que se pueden utilizar tiene un rango de 1 a 40, este elemento se lo ubicará en un QComboBox como se observa en la Figura 2.13.

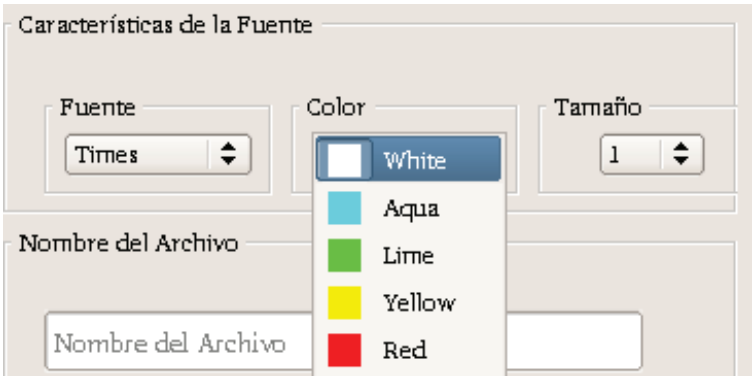

Figura 2.12 QComboBox “Color de texto”

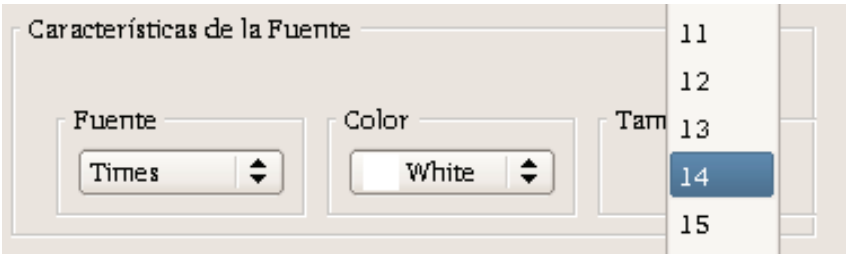

Figura 2.13 QComboBox “Tamaño del texto”

Con la utilización de QComboBox no se pueden ingresar otros valores que no consten en este elemento evitando que el usuario que cometa posibles errores que se dieran al escoger parámetros no soportados por el consumidorRSS, con esto se evitan las validaciones. Con QComboBox se tiene una presentación amigable y llamativa para el usuario, un ejemplo de esto es cuando se llama al color de la fuente, en donde se presentará una paleta de los colores que se pueden utilizar como se observa en la Figura 2.12.

Una vez integrada la aplicación diseñada con Qt Creator se añadieron nuevos requisitos ya que cuando se acopló con *Composer* NCL se estableció que se deben añadir más opciones para que el consumidorRSS cree automáticamente los diferentes elementos que interactúan en un programa NCL como son: media, área, descriptor y conectores base, todos estos elementos se agregarán con sus propiedades respectivas. En la Figura 2.14 se pueden observar los cambios realizados al consumidorRSS.

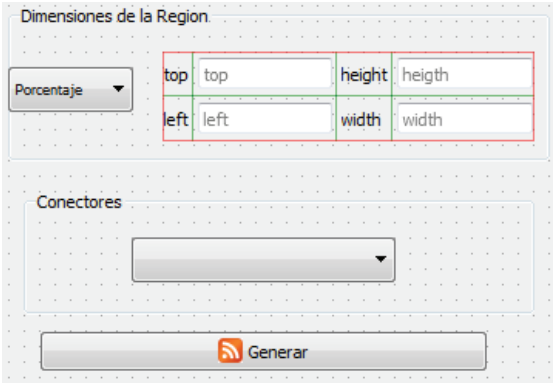

Figura 2.14 Complemento de la interfaz gráfica del consumidorRSS

Básicamente se utilizaron dos grupos de contenedores que poseen esta información, en el uno las dimensiones de la región y en el otro los conectores de la aplicación.

En el grupo “Dimensión de la Región” se agregó un QComboBox que permite escoger dos opciones: Porcentaje y Píxeles. Si se escoge Porcentaje en las propiedades *top*, *left*, *heigth* y *width* se pueden ingresar valores decimales y un valor máximo de cien. Si por el contrario, se escoge Píxeles solo se pueden ingresar valores enteros. Todos estos valores deben ser validados para que no exista ningún problema cuando se ingrese al archivo NCL, ya que como se mencionó anteriormente este *plugin* no está enfocado para personas que sepan programación NCL, sino para personas comunes o diseñadores que quieran implementar esta aplicación a su diseño de software sin cometer errores. En la Figura 2.15 se puede observar su presentación.

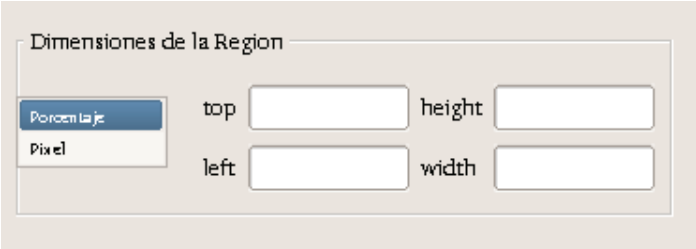

Figura 2.15 Propiedades del área

Para el grupo de “Conectores” se utilizó también un QComboBox en el que se agregaron los conectores que pertenecen a “defaultConnBase” que está integrado en

Composer NCL. Además se programó que el primer elemento que se presente en este QComboBox sea “Inicio Automático”, el cual, al ser seleccionado, agrega un *port* en el *body* del documento NCL que está asociado al elemento media que se generará cuando entre en funcionamiento el comsumidorRSS, lo que implica que se inicie de forma automática sin la necesidad de otro elemento media. En la Figura 2.16 se puede observar lo implementado.

Figura 2.16 QComboBox “Conectores”

Cuando se ejecute el consumidorRSS, se generará un archivo NCL el cual contendrá un elemento media asociado al *body* de la estructura NCL de dicho archivo, además de esto el consumidorRSS generará un elemento área, con las dimensiones establecidas en la interfaz de usuario. Esta área se utilizará con el descriptor del elemento media para especificar en qué lugar de la pantalla se presentarán los resultados obtenidos con el consumidorRSS. Por último se escoge cuándo se presentará la información del *feed* RSS en la aplicación, para lo cual se deben agregar los conectores en el *body* del archivo NCL.

Es posible que varios de los conectores requieran de otros elementos media para interactuar entre ellos, por lo que no es posible automatizar todos los conectores, por lo que el usuario deberá especificar ciertas propiedades de forma manual. El consumidorRSS valida que exista el archivo con los conectores base, en caso de que no exista, el consumidorRSS creará el archivo con dichos conectores.

La lista de conectores base está filtrada para que solo se presenten los conectores que tengan en sus elementos la palabra *start*, porque con este elemento se puede indicar cuándo se deberá presentar (iniciar) la información del *feed* RSS. En la Figura 2.17 se puede observar la presentación final del consumidorRSS con todos los cambios ya realizados para la integración con *Composer* NCL.

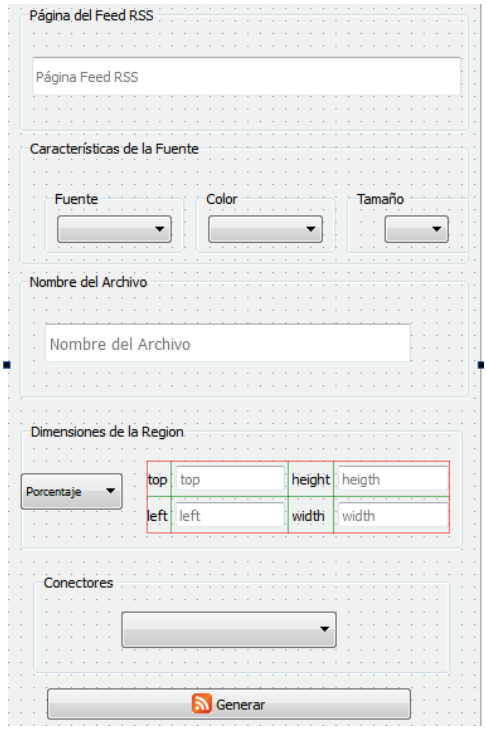

Figura 2.17 Presentación final del consumidorRSS

2.4.7 PRUEBAS DEL PLUGIN CON COMPOSER NCL

A continuación se presentan las pruebas realizadas con el *plugin* consumidorRSS al ser integrado con la aplicación *Composer* NCL.

En la Figura 2.18 se observa una captura de pantalla en la que se presenta el consumidorRSS y el *plugin* vista estructural, en este caso particular se observa un elemento media llamado `ejemplotesis` y un *port*. Únicamente aparecen estos dos elementos debido a que se escogió `InicioAutomatico` en los conectores.

En la Figura 2.19 se observa una captura de pantalla en la que se presenta el consumidorRSS y el *plugin* vista de esquema, lo que se puede apreciar en el *plugin* vista de esquema es que cuenta con un *body* que contiene dos elementos: un elemento media llamado `ejemplotesis` y un *port* llamado `pejemplotesis`. El *head* cuenta con una región base la cual tiene como hijo un área la cual se llama `rgejemplotesis`, un descriptor llamado `descejemplotesis` y un conector base,

tanto los elementos del *head* como los del *body* se generaron automáticamente desde el consumidorRSS.

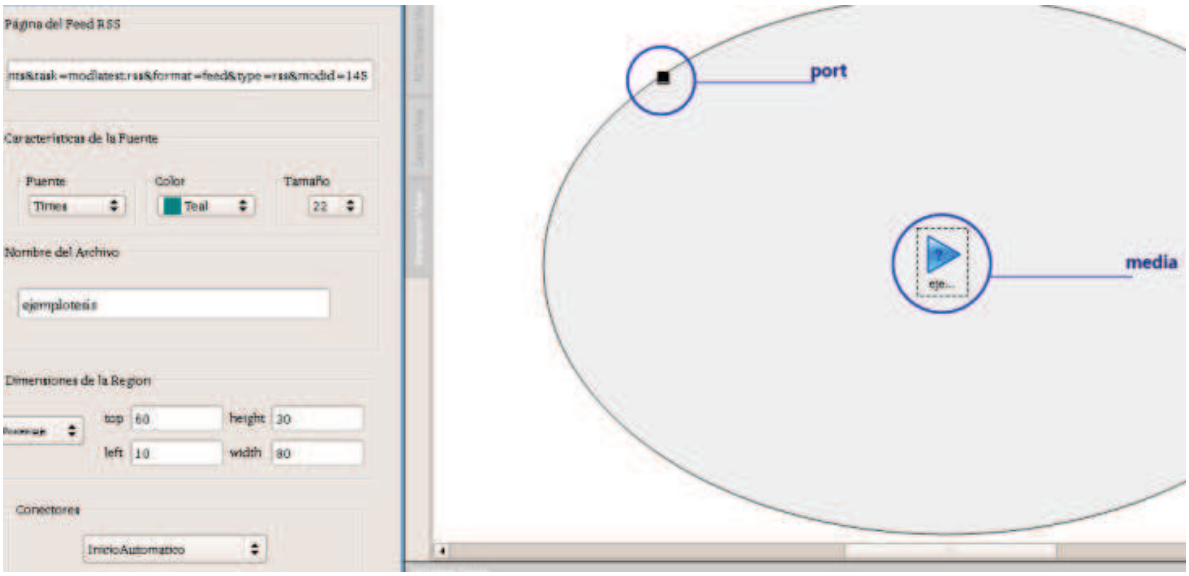

Figura 2.18 ConsumidorRSS con vista estructural

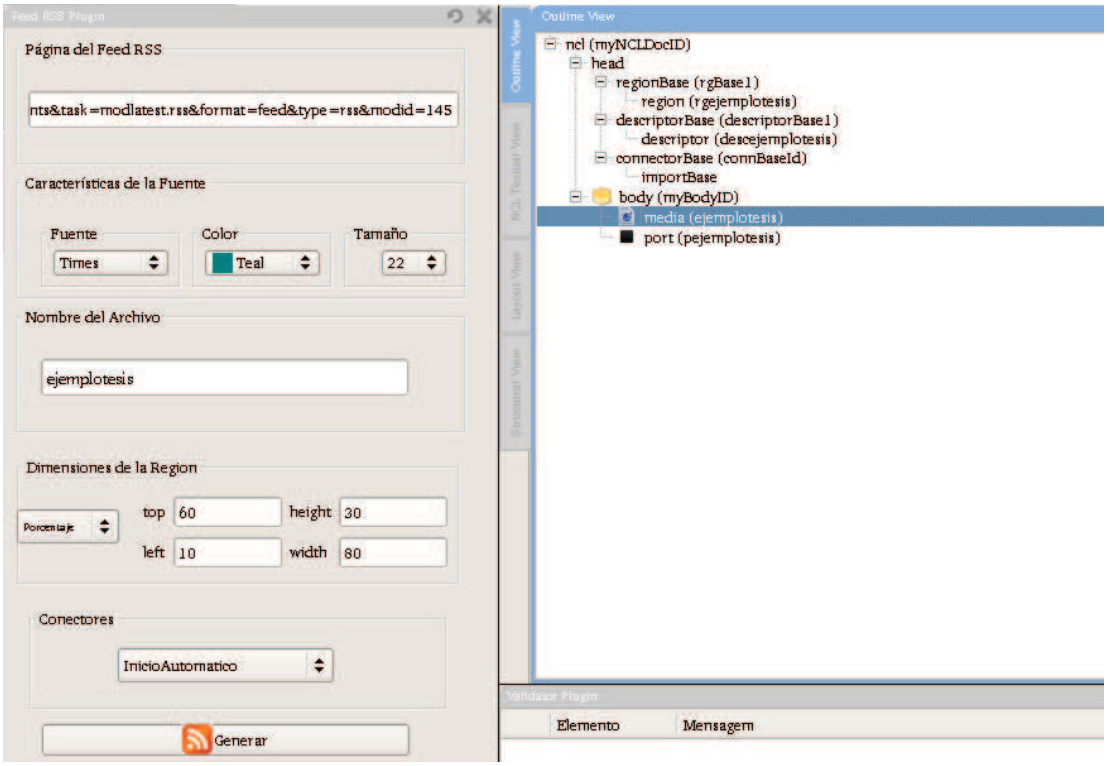

Figura 2.19 ConsumidorRSS con vista de esquema

En la Figura 2.20 se observa una captura de pantalla en la que se presenta el consumidorRSS con el *plugin* vista *layout* y el *plugin* vista de propiedades, en esta figura se aprecia en el lado izquierdo la interfaz de consumidorRSS, en el medio la interfaz vista *layout* y en el lado derecho la interfaz vista de propiedades.

A través del consumidorRSS se especificó las dimensiones del área que son: *top* 60%, *left* 10%, *width* 80% y *heigth* 35%, se puede apreciar que en la vista *layout* se generó la región cumpliendo con los datos establecidos, también se puede apreciar que en el *plugin* vista de propiedades se puede observar estas características. Además se puede apreciar que la región se le ha dado un nombre de `rgejemplotesis`, el nombre se lo da en función del archivo agregando el parámetro “rg” al principio.

En la Figura 2.21 se observa el consumidorRSS y el *plugin* vista textual, en este documento NCL se puede apreciar que se han generado automáticamente con el consumidorRSS las regiones, descriptores, conectores, elementos media y el *port* de acuerdo a lo especificado.

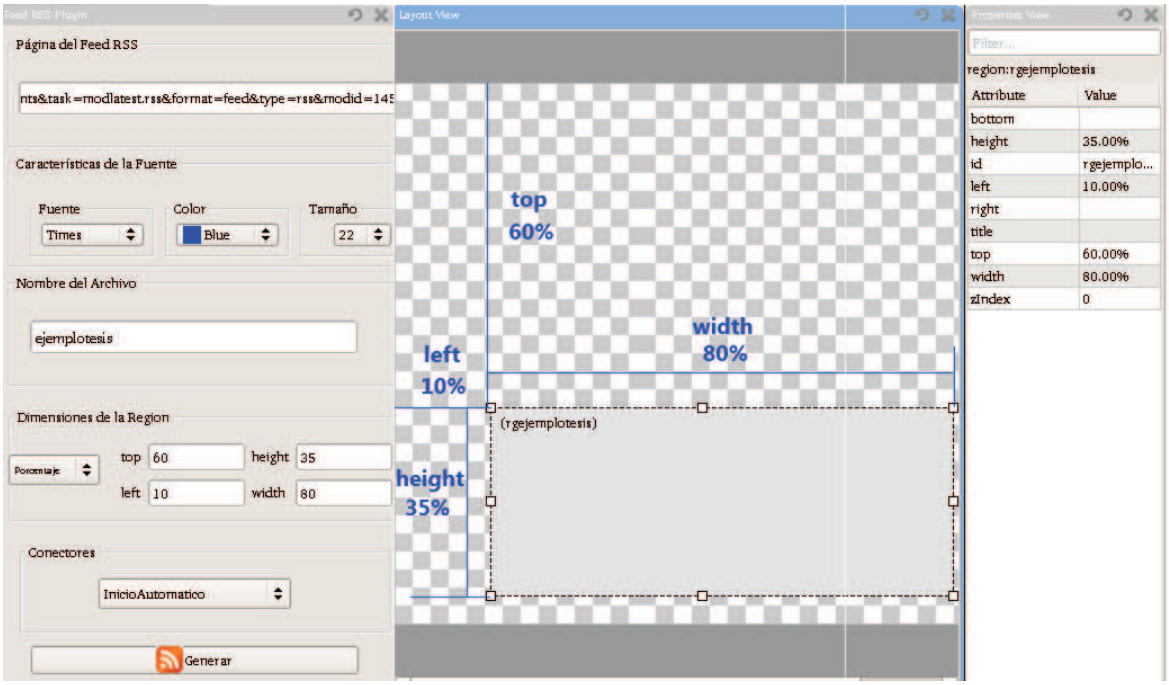

Figura 2.20 ConsumidorRSS con vista *layout* y vista de propiedades

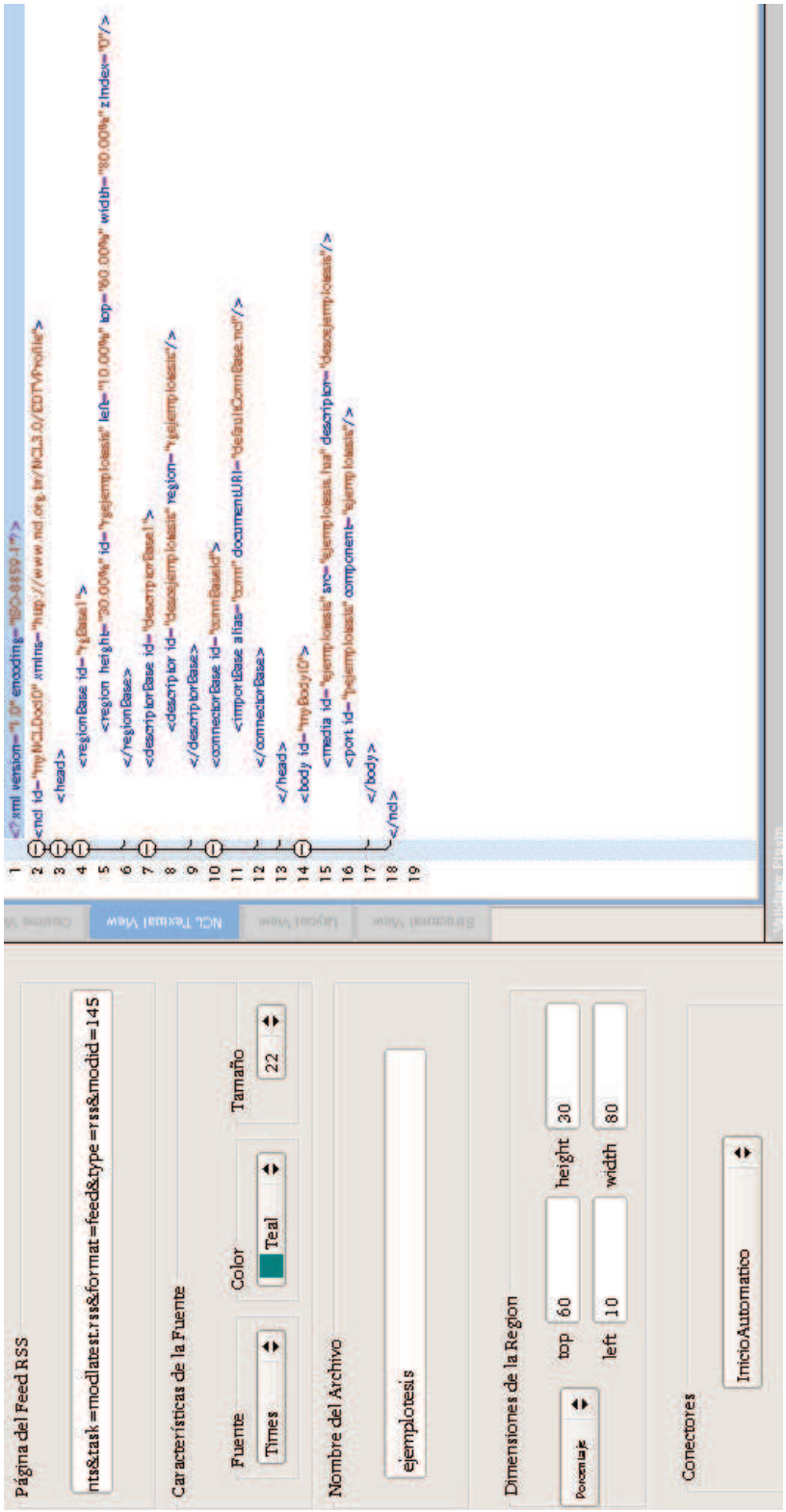

Figura 2.21 ConsumidorRSS con vista textual

Las pruebas se realizaron con el simulador Ginga. En la Figura 2.22 se observa cuando la aplicación intenta buscar la información del *feed* RSS para lo cual despliega el mensaje “Buscando *feed* RSS”.

En la Figura 2.23 se observa la primera noticia después de haber procesado la información con el código NCL. Esta noticia se presentará con los valores que se escogieron anteriormente como el color, tamaño y tipo de fuente, para este ejemplo se utilizó el *feed* RSS de la página de la Escuela Politécnica Nacional.

Figura 2.22 Aplicación diseñada con consumidorRSS ya en funcionamiento

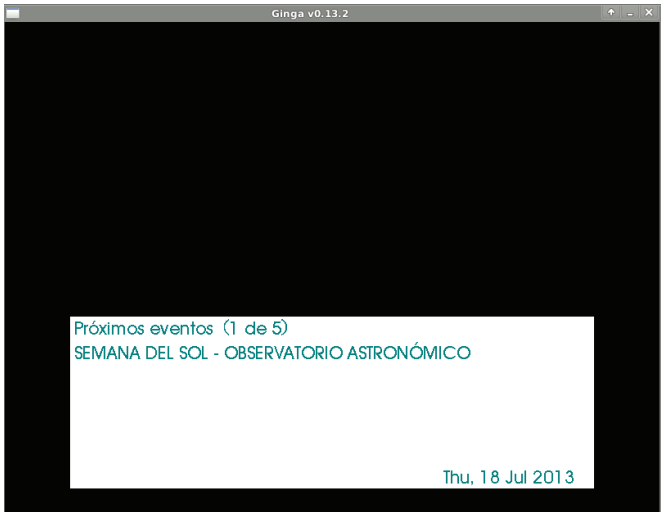

Figura 2.23 Aplicación con la primera noticia

REFERENCIAS BIBLIOGRÁFICAS

[1] RSS 2.0 at Harvard Law. *http://cyber.law.harvard.edu/rss/rss.html*. Consultado en Marzo 2013.

[2] Really Simple Syndication. *http://www.reallysimplesyndication.com/*. Consultado en Marzo del 2013.

[3] What is RSS?. *http://www.hardware-revolution.com/resources/rss-2/*. Consultado en Marzo 2013.

[4] Icai, Pablo J. "Really Simple Syndication (RSS)". http://es.scribd.com/doc/56392931/Sistemas-Distribuidos-RSS. Consultado en Marzo 2013.

[5] Eliomark, Pinzón Rondón. "*AGGREGATORS"*. http://www.slideshare.net/eliomark/diapositivas-rss-eliomark. Consultado en Marzo 2013.

[6] Introduction to RSS. *http://www.webreference.com/authoring/languages/xml/rss/intro/index.html*. Consultado en Marzo 2013.

[7] What Is RSS. *http://www.xml.com/pub/a/2002/12/18/dive-into-xml.html.* Consultado en Febrero 2013.

[8] What Is RSS? RSS Explained. *http://www.whatisrss.com/*. Consultado en Enero del 2013.

[9] Varios autores. "*HD Magazine Nro 5"*. http://www.hdmagazine.org/. Consultado en Enero del 2013.

[10] Marcelo Hernán, Schenone. "*Diseño de una Metodología Ágil de Desarrollo de Software"*. http://materias.fi.uba.ar/7500/schenone-tesisdegradoingenieriainformatica.pdf. Consultado en Enero del 2013.

[11] Penades, Maria Carmen. *"Metodologias Aguiles en el desarrollo de Software"*. http://www.willydev.net/descargas/masyxp.pdf. Consultado en Febrero 2013.

[12] ¿Qué es el método Kanban para la gestión de proyectos?. *http://www.javiergarzas.com/2011/11/kanban.html*. Consultado en Febrero 2013.

[13] DESARROLLO DE KANBAN. *http://www.desarrolloweb.com/articulos/desarrollo-agil-kanban.html*. Consultado en Febrero 2013.

[14] Henrik Kniberg & Mattias Skarin . *"Kanban y Scrum – obteniendo lo mejor de ambos".* http://www.proyectalis.com/documentos/KanbanVsScrum_Castellano_FINAL-printed.pdf. Consultado en Febrero 2013.

[15] Exemplo 6 - Consulta ao Google. *http://www.telemidia.puc-rio.br/~francisco/nclua/tutorial/exemplo_06.html*. Consultado en Marzo 2013.

[16] Módulo canvas. *http://www.telemidia.puc-rio.br/~francisco/nclua/referencia/canvas.html*. Consultado en marzo de 2013.

[17] Paucar Curasma, Ronald. *"Análisis y Modelamiento de las Técnicas de Canal de Retorno e Interactividad para el Estándar de Televisión Digital Terrestre ISDB-T"*. publicado en 10 de Diciembre de 2010

[18] Moreira Gomes, Johnny. *"Utilizando o Lua Xml Parser para leitura simples de arquivos xml em aplicações NCL/Lua"*. http://www.ufjf.br/lapic/files/2010/06/Tutorial_Lua_XML_Parser.pdf. Consultado en marzo de 2013.

[19] Anonimo . *"APRENDA Qt4 DESDE HOY MISMO"*. http://www.choccac.com/attachments/article/11/Aprenda_Qt4_Hoy_Mismo.pdf. Consultado en marzo de 2013.

[20] Blanchette, Jasmin. *"C++ GUI Programming with Qt 4"*. http://grimaldi.univ-tln.fr/Qt/C++-GUI-Programming-with-Qt-4-1st-ed.pdf. Consultado en marzo de 2013.

[21] Pagina oficial de Qt Creator. http://qt-project.org/wiki/category:tools::qtcreator. Consultado en Mayo 2013.

[22] How to: Create an NCL Composer Plugin . http://composer.telemidia.puc-rio.br/en/doc/tutorial/how_to_create_a_plugin_to_ncl_composer. Consultado en Julio 2013

[23] NCL Composer. http://composer.telemidia.puc-rio.br/doc/. Consultado en Julio del 2013

[24] ABNT NBR 15606-2, *http://www.abnt.org.br/imagens/Normalizacao_TV_Digital/ABNTNBR15606-2_2007Ing_2008.pdf*. Consultado en enero 2013

CAPÍTULO 3

DESARROLLO Y PRUEBAS DE APLICACIONES INTERACTIVAS

3.1 INTRODUCCIÓN

En este capítulo se presenta un análisis del contenido de las aplicaciones, también se explicará el desarrollo de una aplicación con el código implementado, se presentan los resultados de las pruebas realizadas tanto en equipos reales como en simuladores y también se presentan los resultados de las encuestas MOS.

En los últimos años en Ecuador se han realizado cambios relevantes en la educación superior como por ejemplo la categorización de las universidades; las becas para estudios tanto para pregrado como postgrados, que en número han aumentado significativamente; el Sistema Nacional de Nivelación y Admisión (SNNA) que en este momento se encarga del ingreso a las universidades; entre otras. Como toda esta información puede ser decisiva para los ecuatorianos se la debe difundir adecuadamente a toda la población y siendo la televisión una indispensable fuente de información y entretenimiento en estos días, se han desarrollado aplicaciones interactivas que permitirán cumplir con este objetivo.

3.2 ANÁLISIS DEL CONTENIDO DE LAS APLICACIONES

A continuación se realizará una descripción de los requerimientos para cada una de las aplicaciones desarrolladas.

3.2.1 CATEGORIZACIÓN DE LAS UNIVERSIDADES DEL ECUADOR [1], [2], [3]

En Ecuador, el CONEA (Consejo Nacional de Evaluación y Acreditación) realizó la categorización de las universidades basándose en el mandato constitucional número

14 que tiene como título “Evaluación de desempeño institucional de las universidades y escuelas politécnicas del Ecuador”. El CONEA clasificó las universidades en cinco categorías desde la categoría A hasta la E, según los siguientes criterios: “academia, estudiantes y entorno de aprendizaje, investigación y gestión interna”; las universidades que pertenecen a la categoría A se presentan en la Tabla 3.1, las universidades que corresponden a la categoría B se exhiben en la Tabla 3.2, en la Tabla 3.3 se presentan las universidades que pertenecen a la categoría C, en la Tabla 3.4 se exhibe las universidades que corresponde a la categoría D y como las universidades que pertenecían a la categoría E fueron cerradas no se las presentan.

En la aplicación se presenta información de las universidades que pertenecen a cada categoría[1], esto se realizó para que tanto los aspirantes así como las personas que se encuentran cursando sus estudios en dichas instituciones sepan la situación en la que se encuentran las universidades porque para muchas personas el tema de la categorización de las universidades es nuevo, o no dispone de información adecuada sobre el mismo.

Existe mucha información que se puede presentar a los televidentes sobre las universidades, pero solo se expondrá los siguientes temas que se consideraron importantes: el nombre del rector, para que los televidentes sepan a quien deben dirigirse para realizar los diferentes trámites; la ubicación, para que los televidentes sepan cómo llegar a las universidades, ya que muchas personas no conocen donde se encuentran ubicados los centros educativos; la página web oficial de la universidad, para que el televidente pueda investigar y obtener mayor información que la presentada, dado que la aplicación no puede abarcar toda la información que se puede encontrar en una página web; un número telefónico, para que el televidente pueda contactarse directamente con la universidad para obtener mayor información en caso que no disponga del servicio de Internet; una dirección de correo electrónico, con el que el televidente pueda obtener información escrita por parte de un empleado de la universidad.

[1] La categorización se basó en el mandato constitucional número catorce por el cual CONEA el 4 de noviembre del 2009 clasificó las universidades en diferentes categorías.

También es conveniente presentar la oferta académica tanto de pregrado como de postgrado, puesto que no todas las universidades ofertan las mismas carreras. Cabe recalcar que muchas universidades ofertan carreras similares, al presentarlas en la aplicación los televidentes tendrían varias opciones para decidirse por una universidad.

Categoría A	
Escuela Politécnica Nacional	Universidad de Cuenca
Escuela Superior Politécnica del Litoral	Universidad del Azuay
Escuela Politécnica del Ejército	Universidad San Francisco de Quito
Escuela Superior Politécnica del Chimborazo	Universidad Técnica de Ambato
Pontificia Universidad Católica de Quito	Universidad Central del Ecuador
Universidad Técnica Particular de Loja	

Tabla 3.1 Universidades de categoría A

Categoría B	
Universidad Agraria del Ecuador	Universidad Estatal de Bolívar
Universidad Católica de Santiago de Guayaquil	Universidad Nacional de Loja
Universidad de Especialidades Espíritu Santo	Universidad Politécnica Salesiana
Universidad Nacional de Chimborazo	Universidad de Guayaquil
Universidad de las Américas	Universidad Técnica del Norte

Tabla 3.2 Universidades de categoría B

Categoría C	
Universidad Tecnológica Equinoccial	Universidad Naval Morán Valverde
Universidad Técnica L. Vargas Torres	Universidad Técnica de Cotopaxi
Universidad Técnica Estatal de Quevedo	Universidad Técnica de Machala
Universidad Estatal del Sur de Manabí	Universidad Estatal de Milagro
Universidad Internacional del Ecuador	Universidad Católica de Cuenca
Universidad Laica Eloy Alfaro de Manabí	Escuela Politécnica Agropecuaria

Tabla 3.3 Universidades de categoría C

Categoría D	
Universidad Casa Grande	Universidad Tecnológica Empresarial de Guayaquil
Universidad de los Hemisferios	Universidad de Especialidades Turísticas
Universidad Estatal Amazónica	Universidad Tecnológica Indoamérica
Universidad Internacional SEK	Universidad San Gregorio de Portoviejo
Universidad Laica Vicente Rocafuerte	Universidad del Pacífico - E. de Negocios
Universidad Regional de los Andes	Universidad Metropolitana
Universidad Técnica de Babahoyo	Universidad Estatal Península de Santa Elena
Universidad Técnica de Manabí	Universidad Iberoamericana del Ecuador
Universidad Tecnológica Ecotec	Universidad Tecnológica Israel
Universidad de Otavalo	Universidad Politécnica Estatal del Carchi

Tabla 3.4 Universidades de categoría D

Para organizar la información en la aplicación existen varias alternativas definidas en [7], de estas opciones la que resulta más sencilla para el usuario como experiencia vivida son los menús, porque de acuerdo a las leyes de Gestalt [8], específicamente la ley de experiencia, el televidente tiende a relacionar lo que observa con experiencias ya vividas y al haber utilizado menús ya sea en una aplicación de escritorio o en una página web le va a resultar más fácil utilizar la aplicación.

Además de esto, con la utilización de menús se obtiene la siguiente ventaja: mejor organización de la información; se puede presentar en una sola pantalla todas las opciones que brinda la aplicación y su estética mejora notablemente.

La información a presentar en pantalla no debe ser desmesurada porque al televidente le parecerá tediosa, por lo cual únicamente se debe presentar la información resumida para que el televidente pueda asimilarla fácilmente, también se debe tomar en cuenta que el tamaño del texto sea el apropiado para que se lo pueda ver claramente desde una distancia adecuada. Basado en esto se ha establecido que la información sea presentada en varias imágenes.

Inicialmente se planificó que la información que la aplicación presentará sería de todas las universidades, así como su oferta académica tanto de pregrado como de

postgrado y su información general, sin embargo el tamaño final de la aplicación resultó demasiado grande lo cual generó problemas con el envío de la información y con la memoria de almacenamiento del STB, dado que por lo general los fabricantes de STB recomiendan que las aplicaciones no pesen más de 6 MB, por este motivo se replanteó este diseño y se consideró que era conveniente presentar únicamente las universidades de categoría A para demostrar el uso que se les puede dar a las aplicaciones, por lo cual se optó diseñar por la aplicación de la siguiente forma:

- Para las universidades que pertenezcan a la categoría A, se presentará su oferta académica tanto de pregrado como de postgrado, además se incluirá información propia de la universidad.
- A las universidades restantes únicamente se las ubicará en las categorías en las que fueron asignadas.

3.2.1.1 Presentación de la información en la aplicación

En base al análisis realizado, las interfaces para la aplicación se realizaron de la siguiente manera:

La aplicación como ya se mencionó contendrá menús, cada ítem del menú dirigirá a la información relacionada con el ítem seleccionado. En la página principal se utilizará un menú con los siguientes ítems: “Categoría A”, “Categoría B”, “Categoría C” y “Categoría D”, en la Figura 3.1 se puede observar el menú mencionado.

Como se mencionó anteriormente, cuando se seleccionen los ítems “Categoría B”, “Categoría C” y “Categoría D” se presentarán las universidades que pertenecen a cada categoría, como esta información es muy extensa se la dividió en varias imágenes, las mismas que contendrán la siguiente información: en la parte superior aparecerá la categoría seleccionada y a continuación se presentarán cuatro universidades, en la Figura 3.2 se puede observar la división de la información explicada. Utilizando el mismo formato se tendrá que las categorías B y C estarán conformadas por tres imágenes y la categoría D estará conformada por cinco imágenes.

Figura 3.1 Menú principal de la aplicación “Categorización de las Universidades del Ecuador”

Figura 3.2 Distribución de la información de las universidades de la categoría B

Al seleccionar el ítem “Categoría A”, se remplazará el menú principal por cuatro ítems de las universidades que pertenecen a esta categoría, además de esto se presentará una imagen con una explicación sobre la categoría A, como se puede observar en la Figura 3.4. La categoría A consta de once universidades, éstas aparecerán en tres menús, cada uno de éstos estará conformado por cuatro universidades, para desplazarse entre las universidades se utiliza el control remoto, el primer grupo de universidades se pueden apreciar en la Figura 3.3.

Figura 3.3 Menú de la categoría A con las primeras cuatro universidades

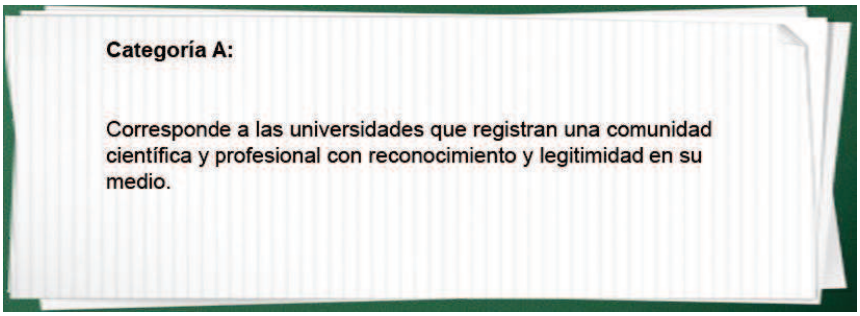

Figura 3.4 Explicación sobre categoría A

Al seleccionar una universidad de la categoría A, se desplegará una imagen con la información general de la misma, como se puede observar en la Figura 3.5.

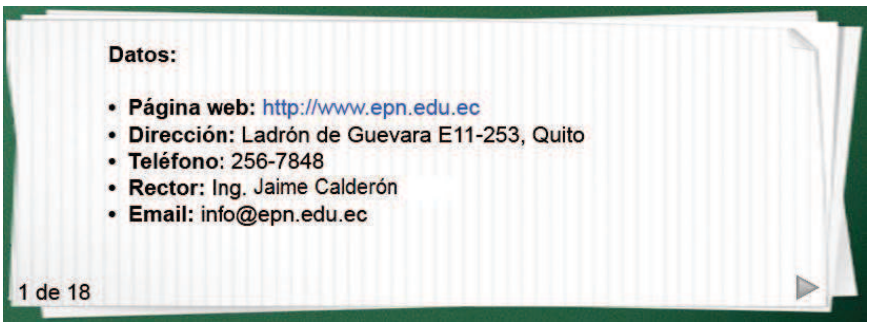

Figura 3.5 Datos generales de la Escuela Politécnica Nacional

Después de presentar la información general de la universidad, aparecerá su oferta académica tanto de pregrado como de posgrado en varias imágenes distribuidas de la siguiente forma: en la parte superior se encuentra la facultad, a continuación se especifica si la oferta académica es de pregrado o de postgrado y por último se presentan cuatro carreras que oferta la universidad, por ejemplo en la Figura 3.6 se observa una primera imagen con la oferta académica de la Escuela Politécnica Nacional, en la que se presenta la facultad de ciencias con sus cuatro carreras.

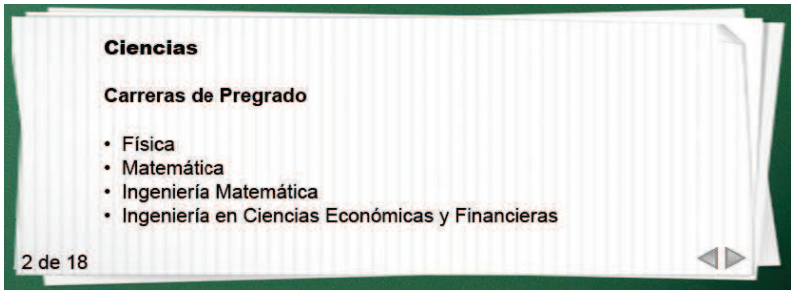

Figura 3.6 Distribución de la oferta académica de la Escuela Politécnica Nacional

Adicionalmente se tiene una ayuda de cómo manipular la aplicación por medio del control remoto. Esta ayuda es importante dado que el usuario aún no está acostumbrado a manipular el control remoto en las aplicaciones interactivas. En la Figura 3.7 se presenta dicha imagen.

Figura 3.7 Imagen de ayuda de la aplicación “Categorización de las Universidades del Ecuador”

3.2.2 BECAS DE UNIVERSIDADES DE EXCELENCIA [4]

El Programa de “Becas Universidades de Excelencia" otorga financiamiento completo para estudiantes residentes dentro y fuera del país que deseen realizar sus estudios de tercer o cuarto nivel en una de las 175 mejores universidades del mundo. Este programa brinda la posibilidad de que personas de todos los estratos sociales puedan postularse a este programa y obtener todos los beneficios que brindan estas becas.

Basado en [4] y en función de varios comentarios realizados por estudiantes que están interesados en obtener una beca o los que ya las han obtenido, se ha escogido para esta aplicación los siguientes temas: “¿Qué estudiar?”, “¿Qué rubros cubre la beca?”, “¿Las becas son reembolsables?”, “Requisitos”, “¿Cómo postular?” y “Documentos habilitantes”. Con estos temas se abarca la mayoría de preguntas que una persona puede plantearse a la hora de escoger una beca. A continuación se explica cada uno de estos temas.

“¿Qué estudiar?”, se escogió ya que muchas personas que buscan una beca en este programa no saben a qué universidades podrían aplicar y que áreas de estudio abarcan. Como la información a presentar es muy extensa ya que las universidades a las que se pueden aplicar con este programa son 175 a nivel mundial distribuidas según su área de estudio, se optó por mencionar solamente la cantidad de universidades y agregar la dirección web del programa de becas para que si el televidente está interesado sepa donde obtener la información.

“¿Qué rubros cubre la beca?”, para la mayoría de postulantes el factor económico es uno de los más decisivos a la hora de escoger una beca, puesto que muchas personas que desean obtener este beneficio no poseen los recursos necesarios para realizar sus estudios en otro país, ya sea para la manutención o para pagar la colegiatura y gastos de viajes. Las becas cubrirán los rubros que se detallan en [4].

“¿Las becas son reembolsables?”, es un tema delicado ya que los futuros becarios necesitan conocer cómo pagarán los montos que les fueron asignados mediante este programa, todas las becas que brinda el gobierno ecuatoriano no se las deben retribuir con dinero sino con condiciones propias que se específica en cada programa. En este tema se explicará cómo se retribuyen estos montos.

“Requisitos”, las personas deben tener una información adecuada de los requisitos que deben cumplir para acceder a este programa porque si no los cumplen no podrán ser beneficiados con una de estas becas y todo el esfuerzo realizado anteriormente será infructuoso. En esta sección se detallan los requisitos que se deben cumplir para aplicar al programa de becas [4].

“¿Cómo postular?”, para que un aspirante pueda postularse a este programa se deben cumplir con una serie de instrucciones que la aplicación las presenta de forma clara y resumida. Los pasos a seguir para que una persona pueda postularse a estas becas se especifican en [4].

“Documentos habilitantes”, existen varios documentos habilitantes que se deben presentar en formatos definidos y siendo estos requisitos imprescindibles en la aplicación se los detallan en forma clara y precisa para que una persona pueda postularse a este programa sin inconvenientes.

Al igual que en la aplicación “Categorización de Universidades”, se utilizarán menús por todos los beneficios y cualidades descritas anteriormente, además de esto los títulos de cada ítem del menú se escogieron en base a las leyes de usabilidad definidas según Steve Krug [7] donde se especifica que entre más corto y obvio sea el título de los ítems, el televidente necesita pensar menos y la aplicación le resulta más atractiva, por esta razón se escogieron los siguientes títulos: “¿Qué estudiar?”, “¿Qué rubros cubre la beca?”, “¿Las becas son reembolsables?”, “Requisitos”, “¿Cómo postular?” y “Documentos habilitantes”.

De la misma manera que en la aplicación “Categorización de Universidades” la información estará dividida en varias imágenes, para no sobrecargar de información en la pantalla y que el tamaño del texto sea adecuado.

3.2.2.1 Presentación de la información en la aplicación

El menú principal contiene 6 ítems que se indicaron previamente, en la Figura 3.8 se puede observar el menú.

En el primer ítem “¿Qué estudiar?” se menciona solamente que existen 175 universidades y si se desea mayor información se recomienda que se dirijan a la página “http://educacionsuperior.gob.ec” donde se encuentra el listado completo de todas las universidades que aplican al programa, en la Figura 3.9 se observa lo mencionado.

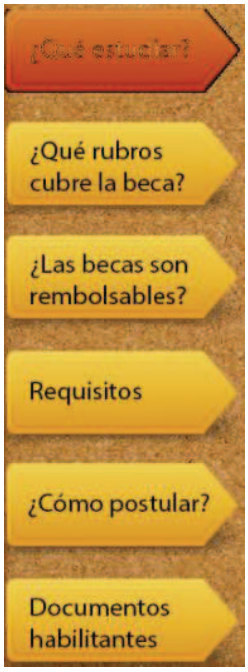

Figura 3.8 Menú de la aplicación “Becas Universidades de Excelencia”

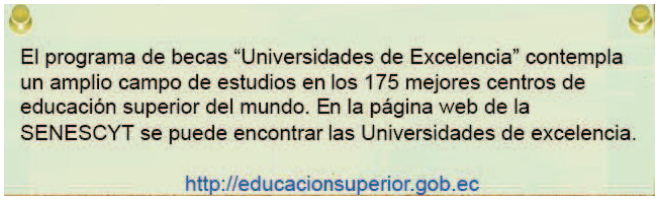

Figura 3.9 Información del ítem “¿Qué estudiar?”

En el segundo ítem “¿Qué rubros cubre la beca?” la información no es extensa, por lo que se la dividió en dos imágenes, en la Figura 3.10 se observa la primera imagen.

Figura 3.10 Primera información del ítem “¿Qué rubros cubre la beca?”

Para el tercer ítem “¿Las becas son reembolsables?” la información se ajustó a una sola imagen dividida en cinco líneas de texto que contendrán los tres puntos, en la Figura 3.11 se observa la información del ítem.

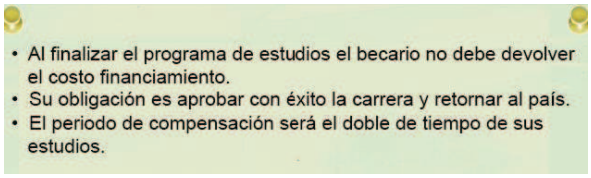

Figura 3.11 Información del ítem "¿Las becas son reembolsables?"

En el cuarto ítem "Requisitos" la información esta detallada en tres imágenes, distribuida en cinco líneas para que el tamaño del texto sea adecuado, en la Figura 3.12 se observa la primera imagen.

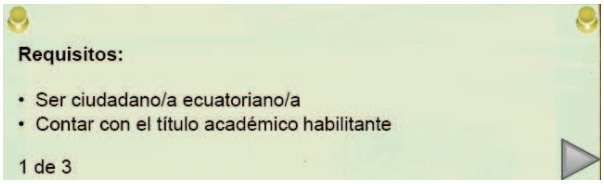

Figura 3.12 Primera imagen de la información del ítem "Requisitos"

El quinto ítem "¿Cómo postular?" contiene información extensa, por lo que se la dividió en tres imágenes, cada imagen tiene cuatro líneas para la distribución de la información, en la Figura 3.13 se observa la primera imagen.

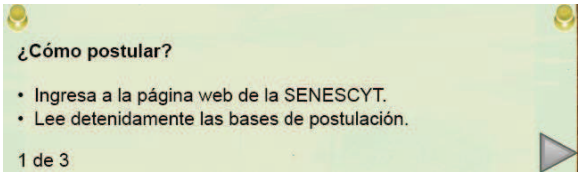

Figura 3.13 Primera imagen con la información del ítem "¿Cómo postular?"

La información del ítem "Documentos habilitantes" se la dividió en cuatro imágenes, cada imagen tiene cuatro líneas para la distribución de la información, en la Figura 3.14 se observa parte del ítem "Documentos habilitantes".

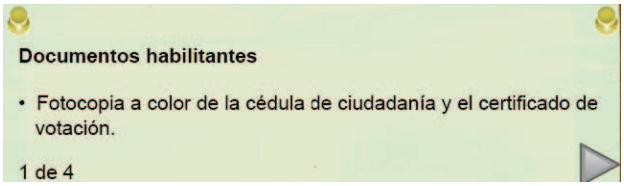

Figura 3.14 Imagen inicial de la información del ítem "Documentos habilitantes"

Además en la página principal se encuentra una imagen de la ayuda para el uso del control remoto dentro de la aplicación, en la Figura 3.15 se presenta una de las imagen utilizada.

Figura 3.15 Imagen de la ayuda del control remoto para la aplicación “Becas Universidades de Excelencia”

3.2.3 SISTEMA NACIONAL DE NIVELACIÓN Y ADMISIÓN (SNNA) [5]

El SNNA tiene como objetivo garantizar la igualdad de oportunidades para el acceso a la educación superior, esto conlleva un nuevo sistema para el ingreso a las mismas, el cual debe ser comunicado adecuadamente dado que los televidentes que se van a informar sobre este tema están definiendo como ingresar a una universidad.

Cuando una persona decide ingresar a la universidad tiene muchas dudas por lo cual se busca con esta aplicación resolver las inquietudes más comunes que un estudiante pueda tener en ese momento como son: “¿Cómo inscribirse?”; “¿Cómo aplicar al ENES?”; “¿Cómo se realiza la postulación?” y “¿Cómo se asignan los cupos?”.

“¿Cómo inscribirse?”, la inscripción para el sistema SNNA es nueva por lo que se requiere informar a los postulantes los requisitos que se deben cumplir para acceder a un centro de educación superior porque si no lo realizan de forma correcta pueden quedar fuera del sistema. Los requerimientos para la inscripción en el SNNA se tomaron de [5].

“¿Cómo aplicar al ENES[2]?”, para ingresar a una universidad se requiere rendir una prueba llamada ENES. Esta prueba es un instrumento para evaluar las aptitudes básicas que la persona necesita para el éxito académico en la universidad, también evalúa cómo los estudiantes analizan y solucionan problemas. En este punto se informará que requerimientos deben cumplir los postulantes para rendir dicha prueba ya que muchos no saben qué materiales están permitidos llevar para rendir la prueba. Los requisitos para aplicar al ENES se detallan en [5].

“¿Cómo se realiza la postulación?”, para rendir la prueba del ENES primero los aspirantes deben postularse, para poder hacerlo correctamente deberán llenar un formulario por medio del internet, en la aplicación se detalla claramente la información que sé deberá llenar en el formato indicado. Para postular a esta prueba se requiere los puntos que se tomaron de referencia en [5].

“¿Cómo se asignan los cupos?”, una vez rendida la prueba del ENES muchos aspirantes no saben cómo se asignarán los cupos, por esta razón se informan los parámetros que se tomarán en cuenta, para que de esta manera los postulantes posean una idea clara de las razones por las cuales les asignaron o no un cupo para estudiar en una universidad. Estos puntos se tomaron de [5].

Esta aplicación utilizará menús al igual que las dos aplicaciones anteriores porque los resultados que se obtienen utilizando este tipo de presentación son del agrado de los televidentes como se explicó anteriormente. Además se escogió los títulos de los ítems que van a conformar el menú según las leyes de usabilidad definidas según Steve Krug [7] que se mencionaron en la aplicación “Becas Universidades de Excelencia” como resultado de esto se tomaron los siguientes títulos: “Inscripción”, “Aplicación del ENES”, “Postulación” y “Asignación de cupos”.

Además de lo anteriormente descrito se debe dividir en varias imágenes la información que se desea presentar debido a que al televidente no le agrada ver información excesiva en una sola pantalla. Adicionalmente se debe tomar en cuenta

[2] Examen Nacional para la Educación Superior

el tamaño del texto porque si es demasiado pequeño el televidente no lo va a poder observar desde una distancia adecuada.

3.2.3.1 Presentación de la información en la aplicación

El menú que utilizará la aplicación consta de los cuatro ítems antes mencionados, en la Figura 3.16 se observa dicho menú.

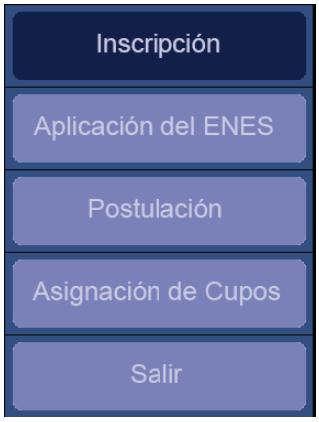

Figura 3.16 Menú de la aplicación “Sistema Nacional de Nivelación y Admisión”

Al seleccionar el primer ítem “Inscripción” aparecerá la información en tres imágenes, en la Figura 3.17 se observa la primera imagen que contiene la información del ítem.

Figura 3.17 Primera imagen que contiene la información del ítem “Inscripción”

En el segundo ítem “Aplicación del ENES” solo se presentarán los requisitos para rendir la prueba, cada elemento está relacionado con una imagen, por ejemplo cuando se está mencionando que se debe presentar con media hora de anticipación en el recinto, se agregó la imagen de un reloj, en la Figura 3.18 se observa la información del ítem “Aplicación del ENES”.

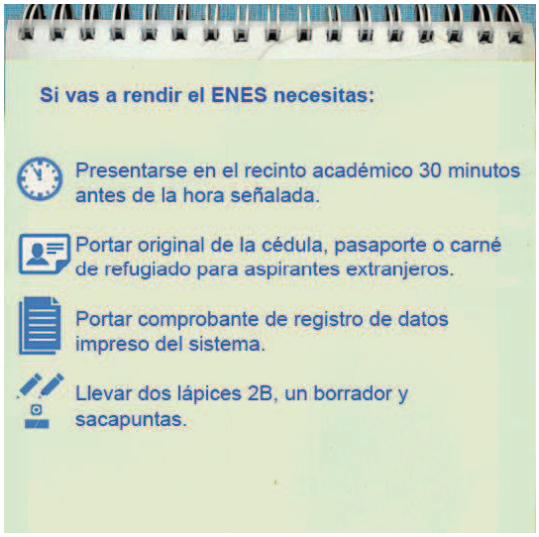

Figura 3.18 Imagen que contiene toda la información del ítem “Aplicación del ENES”

En el tercer ítem “Postulación” la información está dividida en tres imágenes, para enriquecer la parte gráfica se agregaron imágenes que ayuden al usuario, por ejemplo en la imagen que detalla el día que se puede ingresar a la aplicación web que está relacionado con el noveno número de la cédula, se usó una imagen con las fechas de ingreso y un gráfico especificando el número de la cédula, en la Figura 3.19 se observa la información con las imágenes que se han mencionado.

En el cuarto ítem “Asignación de cupos”, la información está en una sola imagen ya que solo se presentan los tres parámetros que se toman en cuenta para la asignación de cupos. En este ítem también se agregó una imagen que explica que la base mínima del puntaje obtenido no siempre asegura un cupo en la carrera deseada, en la Figura 3.20 se observa la información de dicho ítem con su imagen respectiva.

Figura 3.19 Primera imagen que contiene la información del ítem “Postulación”

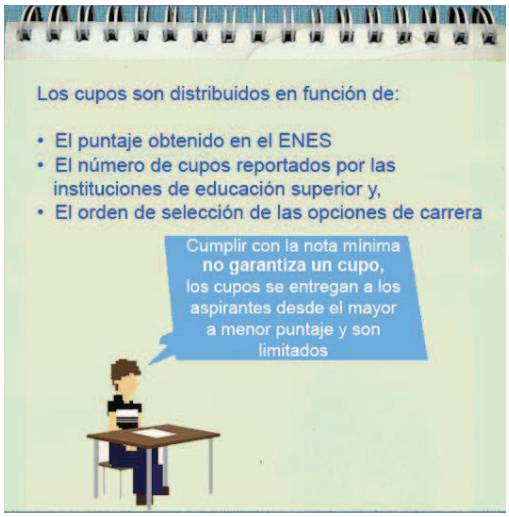

Figura 3.20 Imagen que contiene la información del ítem “Asignación de cupos”

En la página principal del menú de inicio no se agregó ayuda para el uso del control remoto en la aplicación. Al seleccionar un ítem se presentará la información del mismo con una ayuda para el control remoto dentro del ítem, en la Figura 3.21 se observa una de las imágenes de ayuda.

Figura 3.21 Imagen con la ayuda para el control remoto

3.2.4 CIUDAD DEL CONOCIMIENTO YACHAY [6]

Esta aplicación es la única que esta sincronizada con un video, por esta razón la información recolectada debe tener relación con el mismo. La aplicación resaltará los puntos importantes del video o brindará una idea más clara de los temas explicados en el mismo, siendo estos temas: “Ubicación”, “Enfoque de la industria”, “Zonificación”, “Oferta académica”, “Empresas” y “Tipos de vivienda”. Cabe recalcar que la información debe ser resumida ya que solo se presentará en pantalla por un tiempo muy corto.

“Ubicación” es importante informar a las personas el lugar donde se edificará la nueva universidad, dado que muchas de ellas no lo conocen. Yachay se encuentra ubicado en el cantón San Miguel de Urcuquí, al noroccidente de la provincia de Imbabura, al norte de Ecuador, en un área de 4.270 hectáreas [6].

“Enfoque de la industria”, es importante informar a todas aquellas empresas que estén interesadas en conocer si van a tener un espacio dentro de la ciudad que tiene enfocada su industria en:

- Tecnologías de la información
- Biotecnología
- Negocios

“Zonificación”, sirve para informar a las personas como van a estar distribuidas en las diferentes áreas que contendrá la nueva ciudad del conocimiento porque existen zonas determinadas según los estudios que se han realizado. Las áreas divididas se especifican en [6].

“Oferta Académica”, la estructura académica de Yachay se basa en cinco áreas del conocimiento que han sido identificadas como prioritarias para el Ecuador. Este tema se lo presenta para que las personas que estén interesadas en estudiar en esta universidad sepan que oferta académica brinda, porque no todas las carreras se van a ofrecer en un inicio, sino las que son de interés nacional. Las licenciaturas e

ingenierías que se ofertarán al inicio en la universidad de Yachay están especificadas en [6].

“Empresa”, servirá para informar a la población que compañías públicas y privadas podrían intervenir en esta ciudad que abre las puertas a muchas empresas privadas que deseen invertir en este proyecto [6]. Las empresas públicas que están relacionadas con este proyecto se obtuvieron del video que se utilizó para la aplicación, estas son:

- Ministerio de Salud Pública
- Ministerio Coordinador de Conocimiento y Talento Humando
- Secretaria Nacional de Educación Superior, Ciencias, Tecnología e Innovación
- Instituto Espacial Ecuatoriano
- Instituto Nacional de Investigación Geológico Minero Metalúrgico
- Instituto Nacional de Meteorología e Hidrología
- Instituto Nacional de Patrimonio Cultural
- Instituto Antártico Ecuatoriano
- Instituto Oceanográfico de la Armada del Ecuador
- Instituto Geográfico Militar
- Instituto Nacional Autónomo de Investigación Agropecuarias

“Vivienda”, sirve para informar los tipos de residencias que se van a ofertar en esta ciudad porque la mayoría de personas que estudiarán en esta universidad necesitan tener un domicilio dentro de Yachay, dado que esta ciudad está lejos de las principales ciudades del país. Los tipos de viviendas de la ciudad de Yachay son:

- Unifamiliares
- Multifamiliares
- Instalación de uso mixto

En esta aplicación solo se presentará la información en un recuadro porque el tiempo de visibilidad del mismo para los usuarios es muy corto, por lo que no es conveniente colocar un menú con mayor información. El diseño que se utilizará para la aplicación

será el empleo de imágenes con una transparencia para que el usuario no pierda la concentración del video puesto que éste es el importante y la aplicación ayudará al video a reforzar las partes importantes o a dar un poco más de información según lo que se requiera.

3.2.4.1 Presentación de la información en la aplicación

Para el diseño de esta aplicación se tomó en cuenta que la presentación de la información de Yachay debe tener un tiempo determinado que esta sincronizada con el video. Por esta razón debe ser lo más resumida que se presentará durante pocos segundos.

Para el primer tema seleccionado “Ubicación” por tiempo y mejor apreciación gráfica se ha optado por la presentación de una sola imagen que contenga un mapa del Ecuador, el cual indica la ubicación donde se encuentra Yachay, este tema tiene una duración en pantalla de quince segundos que arrancan en el segundo treinta del video.

El segundo tema “Enfoque de la industria” consta de tres ítems con toda la información relacionada con el video presentado, este tema tiene una duración de catorce segundos presentado desde el segundo cincuenta y dos.

En el tercer tema “Zonificación” solo se presentarán las zonas con su extensión, ya que en el video se muestran las zonas pero no su tamaño, este tema tiene una duración de diecisiete segundos presentados desde el segundo ochenta y siete.

El cuarto tema “Oferta académica” se dividió en dos partes: Licenciaturas e Ingenierías, en cada una de ellas consta la oferta que tienen al momento, en esta información se puede interactuar porque con los botones del control remoto se puede cambiar la información.

Además de esto se encuentra una imagen relacionada a los estudios con los tiempos que se demoraría una persona en cursarlos, toda esta información será presentada durante treinta segundos, arrancando desde el segundo ciento veinte. El tema

“Empresas” se dividió en dos segmentos: empresas privadas y públicas. Estas empresas no se las van a nombrar sino que se pondrán imágenes que representarán a cada una de ellas. Como las empresas privadas son pocas, solo se presentarán en una imagen, en cambio las públicas se presentarán en dos imágenes, en este tema también se puede interactuar con el control remoto cambiando la información entre las empresas públicas y privadas, esta información se presentará en trece segundos, a partir del segundo ciento cincuenta y dos.

Como en el último tema “Tipos de viviendas” lo presentado es reforzado por el video y además solo contiene tres ítems, este será presentado en una sola imagen que durará doce segundos, iniciando en el segundo doscientos noventa y tres. En esta aplicación no se presentará ayuda, dado que los tiempos de cada tema son muy cortos y el video es lo más importante.

3.2.5 SOFTWARE PARA LA CREACIÓN DE APLICACIONES GINGA NCL

Para el diseño de aplicaciones Ginga se deberían tomar en cuenta dos aspectos:

Para la programación de las aplicaciones se utilizaron los siguientes programas:

- *Composer* NCL [11]
- Eclipse NCL [12]
- Set-top Box Virtual GINGA-NCL [13]
- Ginga4Windows [14]
- OpenCaster [18], [19], [20]
- StreamXpress

Diseño gráfico de las imágenes para las aplicaciones, se recomienda utilizar el siguiente programa:

- Adobe Photoshop CS6 [15]

3.3 APLICACIONES GINGA

En esta sección se describe una de las aplicaciones desarrolladas. El código de las aplicaciones restantes no se presenta debido a que su programación es similar al de esta aplicación que se tomará como ejemplo, pero se los presentarán adjuntos en el Anexo A.

3.3.1 DESARROLLO DE LA APLICACIÓN

A continuación se presenta el desarrollo de la aplicación “Sistema Nacional de Nivelación y Admisión”.

En la Figura 3.22 se presenta la distribución en la pantalla generada con el *plugin* vista *layout* de los elementos media que intervienen en la aplicación, estos elementos son imágenes con la información que se desea presentar, se han establecido las regiones de esta manera para colocar un menú con cinco ítems que se desplegarán en la parte derecha de la pantalla, además en la parte superior del menú existe un área en la que se desplegará una barra informativa.

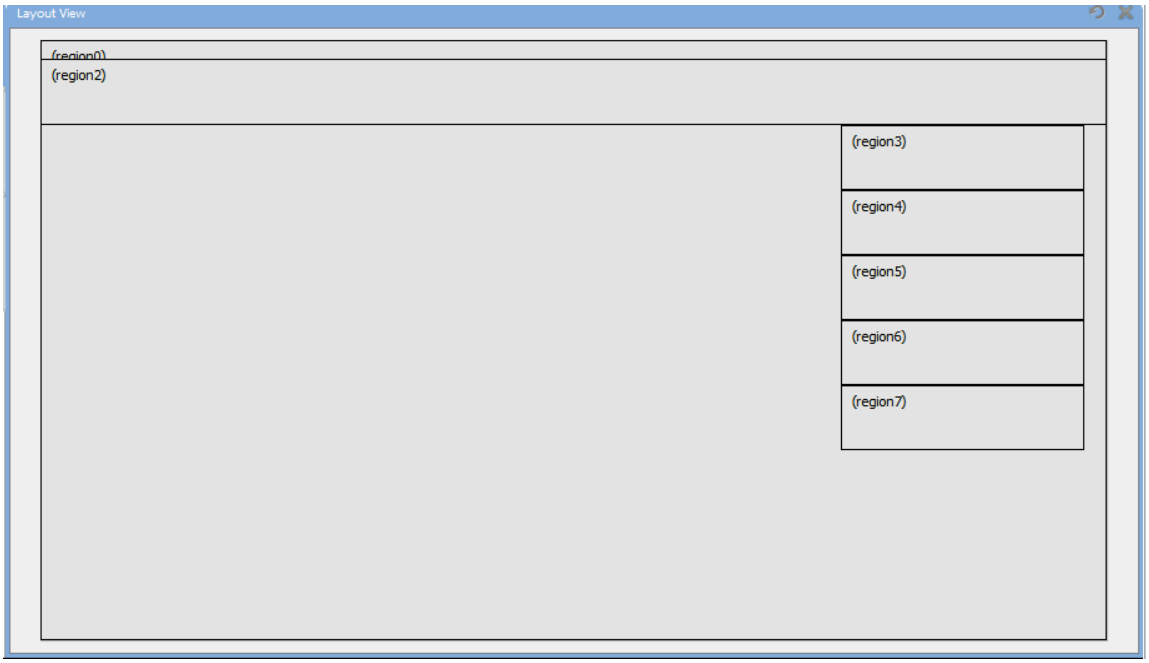

Figura 3.22 Distribución de los elementos en pantalla utilizando el *plugin* vista *layout*

El archivo NCL como se mencionó en el Capítulo 1 tiene que estar formado por un *body* y un *head*. En el *head* se define que se utilizará un archivo donde se encuentran todos los conectores, en el Código 3.1 se presenta dicho código, este código contiene un `alias` con la siguiente cadena de caracteres `conn` que representa a la biblioteca de conectores y en `documentURI` con la siguiente cadena de caracteres `defaultConnBase.ncl` que es el archivo NCL que contiene todos los conectores.

```
<importBase alias="conn" documentURI="defaultConnBase.ncl">
</importBase>
```

Código 3.1 Código para incluir conectores de otro archivo NCL

Además de esto en el *head* se definió un *regionBase* que contiene todas las regiones del documento NCL, en este caso se definieron once regiones. También en el *head* se incluye un *descriptorBase* que abarca todos los descriptores de la aplicación que en total son veintiuno.

La aplicación requiere presentar la señal de la programación recibida por el STB, por esta razón en el *body* se ha colocado un *port* que está asociado a un elemento media llamado `mVideo`, es necesario establecer en el elemento `mVideo` la propiedad `src` con la cadena "`sbtvd-ts://0`", con esta propiedad del elemento `mVideo` la programación normal recibida por el STB seguirá presentando en la aplicación.

Para la presentación del ícono de interactividad se estableció que cuando inicie la aplicación el ícono se va a presentar después de un tiempo determinado, para esto se utilizó el conector `onBeginStart_delay`, este conector permite asociar la presentación de un elemento en particular con un retardo con la presentación de otro elemento, para lo cual se dispone de dos elementos *bind*: `onBegin` y `start`.

Mediante `onBegin` el cual está asociado con un elemento media, el conector se iniciará al ser presentado dicho elemento, produciendo las acciones definidas por el segundo elemento *bind* `start`. La acción definida mediante `start` también asociada con un elemento media permite indicar que elemento se presentará en

pantalla. Adicionalmente el elemento `start` tiene un parámetro `bindParam` que permite especificar el tiempo de retardo, esto permite definir que el elemento media asociado al elemento *bind* `start` se presente luego de un cierto tiempo después de la presentación del elemento media asociado a `onBegin`.

En el Código 3.2 se presenta el conector `onBeginStart_delay` implementado, este conector permite que el ícono de interactividad definido mediante el elemento media `mInteractividad` se presente en la pantalla luego de un segundo de iniciado el elemento media `mVideo`.

```
<link id="link0" xconnector="conn#onBeginStart_delay">
    <bind component="mVideo" role="onBegin">
    </bind>
    <bind component="mInteractividad" role="start">
        <bindParam name="delay" value="1s">
        </bindParam>
    </bind>
</link>
```

Código 3.2 Conector `onBeginStart_delay`

Para mejorar la organización de los elementos que van a interactuar en la aplicación se creó un *context* llamado `ctxSNNA`. Para ingresar a la información de la aplicación se debe de conectar el elemento `mInteractividad` con el *context* `ctxSNNA`, para esto utilizó el conector `onKeySelectionStopStart`, este conector permite que en el momento en que se presente un elemento media y se presione un botón del control remoto, el conector se active y oculte el elemento media, para luego presentar otro elemento NCL.

En el Código 3.3 se presenta el conector `onKeySelecionStopStart` implementado, en el cual se puede apreciar que el elemento media `mInteractividad` se ocultará y se iniciará el *context* `ctxSNNA` cuando el usuario presione el botón “OK” el cual activará el conector, cabe recalcar que la acción de presionar el botón del control remoto viene indicado en la imagen del ícono de interactividad el cual llamará la atención del usuario.

```
<link id="link1" xconnector="conn#onKeySelectionStopStart">
    <bind component="mInteractividad" role="onSelection">
        <bindParam name="keyCode" value="ENTER">
        </bindParam>
    </bind>
    <bind component="mInteractividad" role="stop">
    </bind>
    <bind component="ctxSNNA" role="start">
    </bind>
</link>
```

Código 3.3 Conector `onKeySelecionStopStart`

Una vez iniciado el *context* `ctxBecas` aparecen los elementos media que permiten navegar por la aplicación con toda la información sobre la misma, estos elementos se los presenta mediante un *port* que se encuentran dentro de `ctxBecas` y están asociados a cada elemento media del menú realizado. El menú consta de cinco ítems que son: “Inscripción”, “Aplicación del ENES”, “Postulación”, “Asignación de Cupos” y “Salir”.

Para desplazarse dentro de estos elementos se utilizan descriptores. Cada descriptor dispone de los siguientes atributos: `focusBorderWidth`, para establecer si el elemento va a tener un recuadro; `focusIndex` para establecer el enfoque que va a tener el elemento al presionar los cursores de desplazamiento en el control remoto; `focusSrc` sirve para establecer la dirección de la imagen que va a remplazar a la anterior al momento de desplazar el cursor sobre el elemento seleccionado; `moveDown` y `moveUp`, para realizar el desplazamiento hacia abajo o arriba respectivamente.

Con estos elementos se realiza un menú vertical el cual contiene los cinco elementos media, cada uno de estos elementos contiene un descriptor que tiene en su atributo, `focusBorderWidth` el valor de `0`, el cual determina que no tendrá un recuadro al seleccionar un elemento; en el atributo `focusSrc` se determinó que en cada elemento del menú se agregare la dirección de una nueva imagen que remplazará al seleccionar un ítem del menú; en el atributo `focusIndex` los valores de `1` a `5` respectivamente, y en los atributos `moveDown` y `moveUp` se establece a que elemento saltará el foco al presionar los cursores “DOWN” y “UP” del control remoto.

Al seleccionar un ítem del menú, la aplicación redimensionará el video y mostrará otra página con la información respectiva del ítem seleccionado. El redimensionamiento se lo consigue definiendo la propiedad `bounds` (límites) del elemento media `mVideo`, pero como el elemento `mVideo` se encuentra fuera del *context* `ctxBecas`, se debe crear un nuevo elemento media dentro de `ctxBecas` llamado `mVideoref` para referenciar a `mVideo`, para esto se necesita establecer en `mVideoref` la propiedad `refer` con `mVideo`.

Para realizar el redimensionamiento de la programación normal de la televisión, ocultar un elemento y presentar otro, se utilizó el conector `onSelectionSetNStartNStopN`, este conector permite que en el momento en que se seleccione un elemento media y se presione un botón específico del control remoto, el conector se active y realice las siguientes acciones: cambie la posición y el tamaño de un elemento; puede ocultar varios elementos y puede mostrar varios elementos.

Este conector contiene cuatro elementos *bind*: `onSelection`, `set`, `stop` y `start`. Mediante `onSelection` se especifican las condiciones que deben cumplirse para iniciar este conector, siendo estas: el elemento media debe estar seleccionado, esto se lo realiza especificado mediante el elemento `component` y el botón que se debe presionar en el control remoto es especificado mediante el parámetro `bindParam`.

Mediante `set` se cambiará la posición o el tamaño del elemento media definido por `component`, para esto se emplea la interfaz `bounds` del elemento media, y mediante el parámetro `bindParam` se especifica su nueva ubicación y sus dimensiones usando una cadena con cuatro valores: “izquierda, parte superior, ancho, altura” especificados en porcentajes o valores definidos en píxeles.

Al presionar el botón “OK”, la aplicación cambiará la posición y el tamaño del elemento `mVideoref`; ocultará todos los elementos del menú e iniciará un *context* con el contenido propio del ítem seleccionado, dentro de `ctxBecas` existe cuatro *context* que son: `ctxENES`, `ctxInscripcion`, `ctxPostulacion` y `ctxCupo`.

En el Código 3.4 se presenta el conector `onSelectionSetNStartNStopN` implementado, en este caso se seleccionó el ítem “Inscripción”, además de lo mencionado anteriormente este conector iniciará su propia información que se encuentra en el *context* `ctxInscripcion`.

```
<link id="link3" xconnector="conn#onSelectionSetNStartNStopN">
    <bind component="m0" role="onSelection">
        <bindParam name="keyCode" value="ENTER">
        </bindParam>
    </bind>
    <bind component="m6" interface="bounds" role="set">
        <bindParam name="var" value="57.36%,8%,39%,39%">
        </bindParam>
    </bind>
    <bind component="mMenu1" role="stop">
    </bind>
    <bind component="mInscripcion" role="stop">
    </bind>
    <bind component="mENES" role="stop">
    </bind>
    <bind component="mPostulacion" role="stop">
    </bind>
    <bind component="mCupo" role="stop">
    </bind>
    <bind component="mSalir" role="stop">
    </bind>
    <bind component="ctxInscripcion" role="start">
    </bind>
</link>
```

Código 3.4 Conector `onSelectionSetNStartNStopN`

Como la información que se quiere manipular en los *contexts*: `ctxENES`, `ctxInscripcion`, `ctxPostulacion` y `ctxCupo` es demasiado extensa se la dividió en diferentes imágenes para no sobrecargar de información desplegada en la pantalla para el televidente y también para que el tamaño del texto sea el adecuado.

Para intercambiar las imágenes que contiene la información desplegada en la pantalla se escogió el conector `onKeySelecionStopStart` ya explicado, lo particular de este conector es que se utilizaron los cursores “LEFT” y “RIGHT” del control remoto para activar la condición de funcionamiento del mismo. En el Código 3.5 se presenta el conector antes mencionado, lo particular de este código es que utiliza el cursor derecho para intercambiar las imágenes que contienen la información.

```
<link id="link116" xconnector="conn#onKeySelectionStopStart">
    <bind component="mlibreta" role="onSelection">
        <bindParam name="keyCode" value="CURSOR_RIGHT">
        </bindParam>
    </bind>
    <bind component="mlibreta" role="stop">
    </bind>
    <bind component="mlibreta2" role="start">
    </bind>
</link>
```

Código 3.5 Conector `onKeySelecionStopStart` con una variación en los componentes para utilizar los cursores del control remoto

Para salir del *context* seleccionado y seleccionar otro ítem del menú principal se utilizó el conector `onKeySelecionNStopNStartN`, este conector tiene un funcionamiento similar a `onKeySelecionStopStart` pero la diferencia es que permite ingresar más de un elemento *bind* tanto para ocultar o para activar. Para activar este conector se debe presionar el botón rojo del control remoto, esto permite que se oculten todos los elementos que están dentro del *context* y se activen los elementos del menú principal, en el Código 3.6 se presenta dicho conector.

```
<link id="link4" xconnector="conn#onKeySelectionNStopNStartN">
    <bind component="mManto1" role="onSelection">
        <bindParam name="keyCode" value="RED">
        </bindParam>
    </bind>
    <bind component="mManto1" role="stop">
    </bind>
    <bind component="mlibreta" role="stop">
    </bind>
    <bind component="mprueba" role="stop">
    </bind>
    <bind component="mlibreta2" role="stop">
    </bind>
    <bind component="media3" role="start">
    </bind>
    <bind component="media1" role="start">
    </bind>
    <bind component="media2" role="start">
    </bind>
</link>
```

Código 3.6 Conector `onKeySelecionNStopNStartN`

El elemento `mVideoref` esta redimensionado y se lo tiene que volver a su tamaño y posición original, el conector que realiza esto es `onKeySelectionSet`. Este conector permite especificar que en el momento en que se presente un elemento

media y se presione un botón específico, el conector se active y puede cambiar su posición y tamaño de un elemento media. Este conector contiene dos elementos *bind*: `onSelection` y `set`, cuyo funcionamiento ya se describió anteriormente. Este conector funciona de la siguiente manera: al presionar el botón "OK" la aplicación procederá a cambiar la posición y tamaño del elemento `mVideoref`. En el Código 3.7 se observa el código de `onKeySelecionSet` implementado, lo peculiar de este conector que se lo utiliza en todos los *context* de `ctxBecas`.

```
<link id="link5" xconnector="conn#onKeySelectionSet">
    <bind component="mManto1" role="onSelection">
        <bindParam name="keyCode" value="RED">
        </bindParam>
    </bind>
    <bind component="mVideoref" interface="bounds" role="set">
        <bindParam name="keyCode" value="RED">
        </bindParam>
        <bindParam name="var" value="0,0,100%,100%">
        </bindParam>
    </bind>
</link>
```

Código 3.7 Conector `onKeySelecionSet`

En el menú principal se encuentra el ítem "Salir", al escoger este ítem el menú principal desaparecerá y se activará nuevamente el ícono de interactividad, esto se lo realizó con el conector `onSelectionStopNStartN`, este conector permite ingresar más de un elemento *bind* tanto para ocultar o para activar.

En la Código 3.8 se observa el código de `onSelectionStopNStartN` implementado, en la cual se puede apreciar que del menú se seleccionará el elemento `mSalir`, ocultándose la información del menú principal y se iniciará otro elemento media llamado `mIconoref` que hace referencia al elemento `mInteractividad`.

La distribución de los elementos media y su interacción es presentada por el *plugin* vista estructural, lo cual se puede observar en la Figura 3.23.

```
<link id="link2" xconnector="conn#onSelectionStopNStartN">
    <bind component="mSalir" role="onSelection">
        <bindParam name="keyCode" value="ENTER">
        </bindParam>
    </bind>
    <bind component="mIconoref" role="start">
    </bind>
    <bind component="mMenu1" role="stop">
    </bind>
    <bind component="mInscripcion" role="stop">
    </bind>
    <bind component="mENES" role="stop">
    </bind>
    <bind component="mPostulacion" role="stop">
    </bind>
    <bind component="mCupo" role="stop">
    </bind>
    <bind component="mSalir" role="stop">
    </bind>
</link>
```

Código 3.8 Conector `onSelectionStopNStartN`

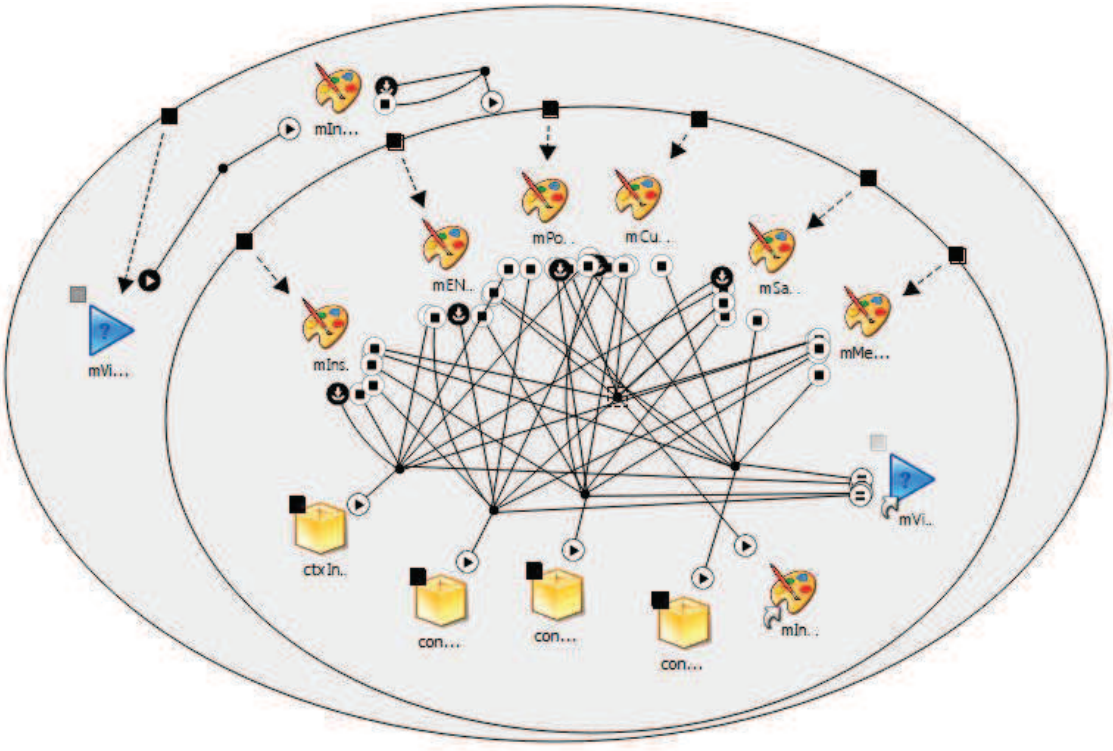

Figura 3.23 Distribución de elementos e interacción en el *plugin* vista estructural

3.3.2 FUNCIONAMIENTO DE LAS APLICACIONES GINGA

En esta sección se presenta el funcionamiento de las aplicaciones Ginga desarrolladas:

3.3.2.1 Categorización de las Universidades del Ecuador

Para el inicio de la aplicación se necesita un ícono que indique que botón del control remoto se debe presionar, este ícono debe estar relacionado con la aplicación a presentar, en este caso se utilizó un birrete, dentro de la figura del birrete esta la palabra “OK” que al televidente le indica que utilice el botón “OK” del control remoto para ingresar a la aplicación.

Se escogió la ubicación del ícono de interacción en la parte superior derecha porque normalmente la información del canal de televisión como su logo viene en la parte superior derecha de la pantalla. Para realizar esta acción se utilizó el conector `onBeginStart_delay` el cual ya fue explicado en la sección anterior. En la Figura 3.24 se observa el inicio de la aplicación.

Figura 3.24 Inicio de “Categorización de las Universidades del Ecuador”

Una vez que se ingresa a la aplicación se redimensiona el tamaño del video, esto se lo realiza con el conector `onKeySelecionSet`, este conector ya fue explicado,

además de esto aparece un menú en la parte izquierda que contendrá los siguientes ítems: “Categoría A”, “Categoría B”, “Categoría C” y “Categoría D”, también se presenta en la parte superior del menú el escudo de la Escuela Politécnica Nacional junto al de la SENESCYT, en la parte central se encuentra el logo del CES, en la imagen principal se encuentra la información del autor de la aplicación así como la información de los directores con la página web del grupo de desarrollo y en la parte inferior se despliega una imagen donde se presenta información sobre el uso del control remoto dentro de la aplicación. En la Figura 3.25 se observa la primera página con todas sus características.

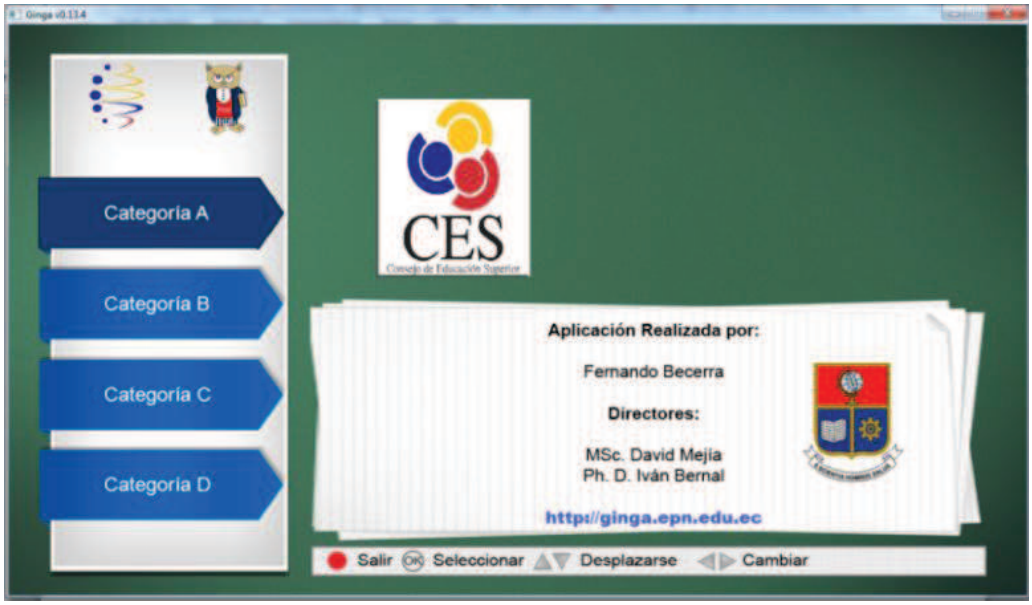

Figura 3.25 Página principal de la aplicación “Categorización de las Universidades del Ecuador”

Al seleccionar el ítem “Categoría A” se ingresa a otra página donde se encuentran a la izquierda las universidades que pertenecen a esta categoría como se observa en la Figura 3.26, en la parte inferior del menú hay la opción de presentar más universidades con el botón de color amarillo del control remoto. En la parte central de la página se encuentra la información de las razones por las cuales las universidades pertenecen a esta categoría.

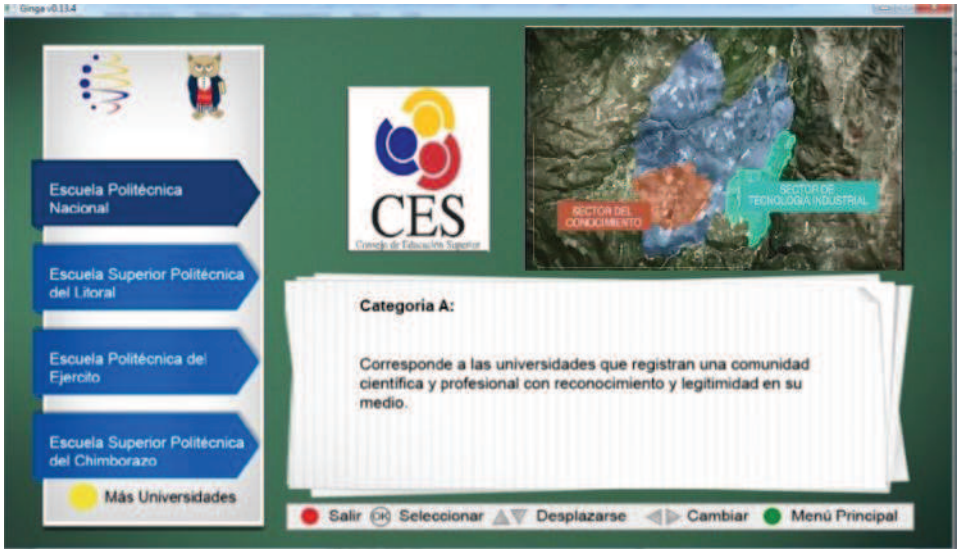

Figura 3.26 Segunda página de la aplicación “Categorización de las Universidades del Ecuador”

Al seleccionar una universidad del menú de la Figura 3.26, por ejemplo el ítem Escuela Politécnica Nacional, su información se desplegará en la parte central de la página, para realizar esto se utilizó el conector `onSelectionStopNStartN` que previamente fue descrito, en la Figura 3.27 se observa el resultado de esta acción. Adicionalmente se puede observar que se agregaron tres imágenes las cuales son: el nombre de la universidad, el escudo y una foto de la institución.

Figura 3.27 Datos del ítem “Escuela Politécnica Nacional”

3.3.2.2 Becas Universidades de Excelencia

En la Figura 3.28 se puede observar el inicio de la aplicación en el simulador, además en esta imagen se puede apreciar que en la parte superior derecha se encuentra el ícono de interactividad el cual está formado por las palabras “Becas” y “OK” que incita al usuario a oprimir el botón “OK” del control remoto para iniciar la aplicación. Para realizar esta acción se empleó el conector `onBeginStart_delay` previamente descrito.

Figura 3.28 Inicio de la aplicación “Becas de Universidades de Excelencia”

Después de presionar el botón OK se redimensiona el tamaño del video, esto se lo realiza con el conector `onKeySelecionSet` que ya se lo manejo anteriormente, además se despliega una página como se presenta en la Figura 3.29, en la que aparece un menú con los siguientes ítems: “¿Qué estudiar?”, “¿Qué rubros cubre la beca?”, “¿Las becas son rembolsables?”, “Requisitos”, “¿Cómo postular?” y “Documentos habilitantes”.

También se presenta en la parte central de la aplicación el escudo de la Escuela Politécnica Nacional junto al de la SENESCYT, además se muestra una imagen relacionada con las becas en la parte inferior de estas dos imágenes.

Esta aplicación es la única que se presenta con el *feed* RSS de la Escuela Politécnica Nacional que mostrará la información de dicha institución en la parte inferior de la imagen, se escogió este *feed* RSS ya que tanto en la página web del programa de becas como en la SENESCYT no existe este elemento y se quería demostrar la funcionalidad del *plugin*.

En la imagen principal se encuentra la información del autor de la aplicación así como la información de los directores con la página web del grupo de desarrollo. También existe un lugar con todas las indicaciones necesarias para utilizar la aplicación con el control remoto.

Figura 3.29 Página principal de la aplicación “Becas de Universidades de Excelencia”

Al seleccionar un ítem de la página que se indica en la Figura 3.29 se presenta la información resumida del mismo, esto se lo realizo mediante el uso del conector `onSelectionStopNStartN` el que previamente fue explicado. En este caso se seleccionó el ítem “¿Qué estudiar?”, cambiando solo la imagen central con la información propia del ítem seleccionado como se aprecia en la Figura 3.30.

Figura 3.30 Contenido del ítem "¿Qué estudiar?"

3.3.2.3 Sistema Nacional de Nivelación y Admisión

En esta aplicación no se especificará los conectores utilizados ya que se los presentó anteriormente en el desarrollo de la aplicación.

Al iniciar la aplicación se presenta la Figura 3.31, en esta imagen se puede observar que lo único que se presenta es el ícono de interactividad el cual se encuentra ubicado en la parte superior derecha de la Figura 3.31 con las siglas "SNNA", además se encuentra la palabra "OK" dentro de un círculo que representa el botón del control remoto.

En esta aplicación se utilizó otro tipo de diseño, en el cual se creó una página que va estar sobrepuesta a la programación normal de la televisión digital. En la Figura 3.32 se observa también que la parte base esta de color negro, esto sucede porque el simulador no tiene recepción de televisión digital porque se está trabajando en un computador. En esta página se encuentra un menú con los siguientes ítems: "Inscripción", "Aplicación del ENES", "Postulación", "Asignación de Cupos" y "Salir".

Figura 3.31 Inicio de la aplicación “Sistema Nacional Nivelación y Admisión”

Además en la parte superior del menú se encuentra una imagen con el escudo de la Escuela Politécnica Nacional con los íconos del SNNA y del SENESCYT. En esta aplicación no se utiliza la ayuda del control remoto para el usuario final porque el menú es fácil de utilizarlo. Esta aplicación es la única de las propuestas en las que existe un ítem en el menú para salir.

Figura 3.32 Página principal de la aplicación “Sistema Nacional Nivelación y Admisión”

Al escoger un ítem se redimensionará la programación habitual de la televisión como se ve en la Figura 3.33 hacia una nueva interfaz en la cual se tendrá la información

de cada ítem de forma concisa y resumida. En la Figura 3.33 se presenta la información del ítem “Inscripción”. Una imagen adicional brinda la información sobre el SNNA como un email y un número telefónico, adicionalmente se presenta el escudo de la Escuela Politécnica Nacional junto al ícono de la SENESCYT.

Figura 3.33 Selección del ítem “Inscripción”

3.3.2.4 Ciudad del conocimiento Yachay

La aplicación Yachay está sincronizada con el video mediante la creación de un flujo único de paquetes de transporte TS.

3.3.2.4.1 Generación de un flujo de paquetes de transporte TS

Para la generación del flujo único de paquetes de transporte TS con contenido de audio, video y datos se debe generar un TS.

3.3.2.4.2 Transport Stream (TS) [17], [21]

Transport Stream es un protocolo de transmisión de audio, video y datos especificados en el estándar MPEG-2, es el formato de compresión más utilizado por codificadores y decodificadores con bajas pérdidas. Para la creación de un TS es necesario crear flujos binarios de vídeo y audio de cada programa que se comprimen

independientemente formando cada uno de ellos un ES[3]. Cada uno de estos ES se estructura en forma de paquetes llamados PES[4]. En la Figura 3.34 se observa los paquetes que intervienen en un TS.

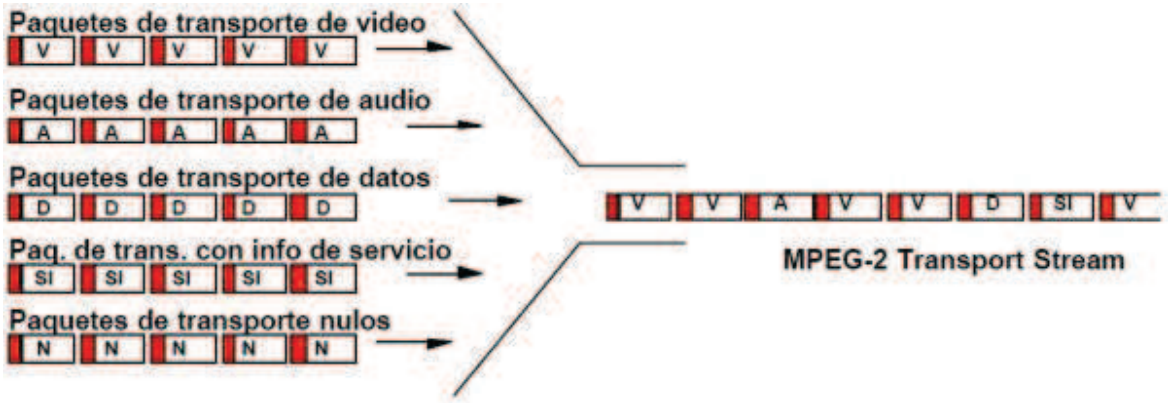

Figura 3.34 Paquetes que intervienen en un TS [21]

Para agregar a la programación, elementos como el nombre del canal o un programa Ginga se utilizan unas tablas llamadas PSI (*Program Specific Information*) que luego se multiplexan con audio y video, el programa Ginga es opcional.

3.3.2.4.3 Tablas PSI [21]

El Múltiplex MPEG-2 *Transport Stream* puede contener varios servicios audiovisuales, cada uno de los cuales está compuesto por uno o varios flujos elementales PES distribuidos en *transport packets*. Estos paquetes a su vez están marcados con un PID (*Packet Identification*) que identifica a qué PES pertenecen.

Sin embargo, para que el decodificador pueda recuperar completamente un servicio a través de los valores de los PID de los paquetes correspondientes es necesario incluir información adicional dentro del flujo de transporte que relacione estos PID con los programas a los que pertenecen. Tal información se denomina *Program Specific Information* (PSI). Las tablas PSI pueden ser:

---

[3] ES (*Elementary Stream*) contiene solo un tipo de datos, por ejemplo audio o video que están definidos por un protocolo como MPEG incluyendo la cabecera de secuencia y todas las subpartes de una secuencia.

[4] PES (*Packetized Elementary Stream*) es una especificación de MPEG-2 que define la realización de ES en paquetes dentro de flujo de MPEG y flujo de transporte MPEG.

PAT (*Program Association Table*) Esta tabla se debe incluir obligatoriamente, contiene una lista completa de todos los programas disponibles en el *transport stream*.

PMT (*Program Map Table*) Esta tabla proporciona detalles del programa y de los flujos elementales (ES) que comprende.

SDT (*Service Description Table*) Esta tabla contiene datos que describen los servicios en el sistema, como por ejemplo: nombres de los servicios, nombre del proveedor y otros parámetros asociados a cada servicio de un mismo *transport stream*.

CAT (*Contitional Access Table*) Esta tabla proporciona detalles de los sistemas de cifrado empleados, así como los valores de los PES que contienen la información del control de acceso condicional.

NIT (*Network Information Table*) Esta tabla está considerada como de datos privados, estos están definidos por el radiodifusor o proveedor de servicios y no por el *transport stream*. Esta tabla proporciona información de la red física usada para transmitir el *transport stream*, como por ejemplo: frecuencias del canal, características de modulación, detalles de redes alternativas disponibles, etc.

Para generar un TS se utiliza un software libre llamado OpenCaster.

3.3.2.4.4 OpenCaster[18], [19], [20]

OpenCaster es un software desarrollado por AVALPA Digital Engineering SRL para generación de *transport streams* MPEG2. Este software es de código abierto, se encuentra bajo prueba y se recomienda instalarlo en Debian pero funciona sobre cualquier versión de Linux.

Este software se lo puede descargar desde la página oficial de AVALPA. Pero el inconveniente con este software es que crea TS para la norma europea DVB-T, por esta razón la Universidad Nacional de la Plata modifico OpenCaster para que pueda crear TS para la norma brasileña ISDB-Tb.

3.3.2.4.5 Generación de TS que incluya una aplicación Ginga [18]

Para la generación de un TS que incluya una aplicación Ginga, se debe tener los siguientes elementos:

- Video
- Tablas PSI
- Aplicación Ginga

Una vez obtenido el video se puede generar los ES de audio y video. A continuación se presenta como se los genera.

Con el comando `ffmpeg` presentado en el Código 3.9 se generó el archivo ES de video, este comando tiene los siguientes parámetros:

- `i` especifica el archivo de entrada para el decodificador, es importante mencionar que puede incluirse la ubicación del mismo.
- `an` permite indicar que se genere únicamente el video sin incluir el audio.
- `vcodec` especifica el códec de video MPEG2 que se emplea para generar el ES de video.
- `f` permite especificar el formato de salida para el archivo ES.
- `S` permite especificar la resolución que tendrá el archivo generado.
- `aspect` define la relación de aspecto.
- `b` permite especificar la tasa de transmisión en bps.
- `maxrate` permite especificar la tasa máxima de transmisión en bps.
- `minrate` especifica la tasa mínima de transmisión en bps.
- `bf` permite especificar el número máximo de b-frames[5] consecutivos. Existen tres valores para este parámetro que son: `0` para desactivar ubicación dinámica, `1` para habilitar una técnica de elección para la colocación rápida

[5] B-frame (*bi-directional frame*) es un método de compresión de video utilizado por el estándar MPEG b-frame se basan en los *frames* que ocurrieron antes y después de ellos, los b-frames solo contienen los datos que han cambiado desde el *frame* precedente o son diferentes de los datos del siguiente *frame.*

dinámica, rápida pero poco precisa y `2` para permitir un modo lento y preciso. Puede ser muy lento si se utiliza con un alto número de b-frames.

- `bufsize` especifica el tamaño del buffer.
- En el último parámetro se debe especificar el nombre del archivo ES de video.

El comando `ffmpeg` presentado en el Código 3.9 permite generar el archivo ES de video para lo cual se especifican los parámetros: él archivo de ingreso es `yachay.mpg` en el que se especifica que no se utiliza audio, el códec para compresión de video es `mpeg2video` el cual es el códec de MPEG-2 video y el formato de salida también es `mpeg2video`, el formato de salida es 720p que sería `1280x720`, la relación de aspecto debe ser `16:9`, en los parámetros `b`, `minrate` y `maxrate` se especificó `400k`, se determinó un mismo valor ya que se empleó CBR[6] el dato de la tasa de transición se la obtiene del video original, en `bf` se utilizó `2` el cual utilizará el método lento y preciso ya que el video que se emplea no es de alta resolución, en el parámetro `bufsize` se utiliza un valor constante para MPEG2 que es `1835008`, este valor se lo obtiene del manual de AVALPA y es vbv_buffer_size*1024*16 = 1’835.008 donde vbv_buffer_size para MPEG2 es 112 y finalmente se indica el nombre del archivo ES `yachay.m2v` donde se almacenará toda la información.

```
ffmpeg –i yachay.mpg -an -vcodec mpeg2video -f mpeg2video -s 1280x720 -aspect 16:9 -b 4000k -minrate 4000k -maxrate 4000k -bf 2 -bufsize 1835008 yachay.m2v
```

Código 3.9 Comando para la generación de ES de video

Para generar el archivo ES de audio se utilizó el comando `ffmpeg` que se presenta en el Código 3.10, este comando utiliza los siguientes parámetros:

- `i` especifica el archivo de entrada para el decodificador, es importante mencionar que puede incluirse la ubicación del mismo.

[6] CBR (*Constant Bit Rate)* la codificación con tasa de bits constante implica que la tasa de salida del codificador de los datos es constante, es muy útil para flujo de datos multimedia con canales de capacidad limitada.

- `av` permite indicar que se genera únicamente el audio sin incluir el video.
- `ac` especifica el número de canales de audio.
- `acodec` se utiliza para especificar el códec MPEG2 de audio que se emplea para generar el archivo ES de audio.
- `f` permite especificar el formato de salida para el archivo ES.
- `ab` permite especificar la tasa de bits de transmisión de audio.
- `ar` permite especificar velocidad de audio, es la frecuencia de muestreo de la señal de audio.
- Por último se debe especificar el nombre del archivo ES de audio.

Para utilizar el comando `ffmpeg` que se encuentra en el Código 3.10 se utilizó el archivo de video `yachay.mpg` del cual en este caso se extrae únicamente el audio, se utiliza un canal de audio estéreo, se utilizó el códec MP2[7] tanto para la compresión de entrada como para la salida de audio, se utilizó `128000`, este dato se obtuvo del video original al igual que `48000` y finalmente se puso un nombre al archivo ES de audio `yachay.mp2`.

```
ffmpeg -i yachay.mpg -vn -ac 2 -acodec mp2 -f mp2 -ab 128000 -ar 48000 yachay.mp2
```

Código 3.10 Comando para la generación de ES de audio

Una vez que se obtuvieron los ES de audio y video el siguiente paso es crear los PES de audio y video.

Para la creación del PES de video se emplea el comando `esvideo2pes` que se presenta en el Código 3.11, el cual acepta como argumento únicamente el nombre del archivo ES y genera un archivo PES como salida. En el ejemplo se puede observar que se ha ingresado como entrada `yachay.m2v` que es el archivo generado anteriormente y con este se obtendrá el archivo `yachayVideo.pes` que es el archivo PES.

[7] MP2 es un formato de compresión de audio con pérdida definida por ISO/IEC 11172-3

```
esvideo2pes yachay.m2v > yachayVideo.pes
```

Código 3.11 Código para la generación del PES de video

Para la creación del archivo PES de audio se utilizó el comando `esaudio2pes` el cual se presenta en el Código 3.12, este comando utiliza los siguientes argumentos:

- El primer argumento permite especificar el archivo ES de video que se desea convertir en PES.
- El segundo argumento especifica el número de muestras por trama.
- El tercer argumento permite especificar la frecuencia de muestreo.
- El cuarto argumento especifica el tamaño de la muestra de audio.
- El quinto argumento permite especificar el valor que se utilizará para la sincronización de audio y video.
- Al final se debe guardar la información creada por el comando en un archivo PES de audio.

El comando `esaudio2pes` que se presenta en el Código 3.12 se utilizó de la siguiente manera: el primer argumento se especificó con `yachay.mp2` que es el archivo ES de audio; tanto el valor `1152`, `48000` y `384` se los obtuvieron del archivo yachay.mp2, estos valores representan: el número de muestras por trama, frecuencia de muestreo y el tamaño de la muestra de audio; el quinto argumento `3600` se lo agregó según recomendaciones del manual de AVALPA para la sincronización del audio y del video y finalmente se agregó el nombre del archivo `yachayAudio.pes` que contiene el archivo PES de audio generado.

```
esaudio2pes yachay.mp2 1152 48000 384 3600 > yachayAudio.pes
```

Código 3.12 Comando para la generación del PES de audio

Una vez generados los PES se deben generar los TS de audio, video y el TS que contiene la aplicación Ginga que se desea multiplexar para generar el flujo único de paquetes de transporte TS.

En el Código 3.13 se presenta el comando `pesvideo2ts` el cual permite especificar los siguientes argumentos para la creación de un archivo TS de video:

- El primer argumento permite especificar el valor del PID, el cual se puede escoger entre 1 a 8191 que será utilizado en las tablas PSI.
- El segundo argumento especifica el número de tramas por segundo del video.
- El tercer argumento permite especificar el valor medio del tamaño del buffer.
- El cuarto argumento especifica la tasa de bits del TS.
- El quinto argumento permite especificar si el video se va a repetir o no, esto se especifica con los valores de 1 o 0 respectivamente.
- El sexto argumento permite definir el archivo PES que se utilizará para la creación del TS de video.
- Finalmente se debe guardar la información generada por el comando en un archivo TS de video.

En el ejemplo se puede observar la creación del TS de video, se ocupó el comando `pesvideo2ts` presentado en el Código 3.13 de la siguiente forma: en el primer argumento se especificó `2065`, este valor se escogió de forma aleatoria dentro de los valores que son permitidos para el uso del PID; en el segundo argumento se puso `25` que representa el número de tramas por segundo, este valor se obtuvo del video original; en el tercer argumento se agregó `112`, este es un valor medio del tamaño del buffer para MPEG2; el cuarto argumento es de `4600000`, valor que se lo obtuvo aumentando un 15% como mínimo de la tasa de transmisión del video original como se especifica en el manual de AVALPA; en el quinto valor se utilizó `0`, con esto se escogió que el video no se repita en un bucle infinito y por último se guardó toda la información creada en el archivo TS de video llamado `yachayVideo.ts`.

```
pesvideo2ts 2065 25 112 4600000 0 yachayVideo.pes > yachayVideo.ts
```

Código 3.13 Comando para la generación del TS de video

El comando `pesaudio2ts` presentado en el Código 3.14 utiliza los siguientes argumentos:

- El primer argumento permite especificar el valor del PID de audio que se encuentra entre 1 y 8191, este valor se utilizará para la generación de las tablas PSI.
- El segundo argumento especifica el número de muestras por trama de audio.
- El tercer argumento permite especificar la tasa de muestreo de audio.
- El cuarto argumento especifica el tamaño de la trama de audio.
- El quinto argumento permite definir si el audio se repetirá o no, los valores que se pueden escoger son 0 para no repetir y 1 para repetir.
- El sexto argumento permite especificar que archivo PES se utilizará para la creación del TS de audio.
- Por último se debe guardar la información creada por el comando en un archivo TS de audio.

Para la generación del TS de audio se utilizó el comando `pesaudio2ts` presentado en el Código 3.14 de la siguiente forma: en el primer argumento se colocó `2075` que específica el PID que se utilizará posteriormente para las tablas PSI; tanto el valor `1152`, `48000` y `384` se los obtuvieron del archivo PES de audio anteriormente generado, estos valores representan el número de muestras por trama, velocidad de muestreo de audio y el tamaño de la trama de audio; en el quinto argumento se colocó `0` porque no se quiere que el audio se repita; en el sexto argumento se colocó `yachayAudio.pes` que es el archivo PES de audio que se desea procesar y finalmente se almacenará toda la información generada en el archivo `yachayAudio.ts`.

```
pesaudio2ts 2075 1152 48000 384 0 yachayAudio.pes > yachayAudio.ts
```

Código 3.14 Comando para la generación del TS de audio

Para generar un TS que incluya una aplicación Ginga se utiliza el *script* `oc-update.sh`, el cual creará un carrusel de objetos[8] como se presenta en el Código 3.15

Los parámetros que se utilizaron para *script* `oc-update.sh` son los siguientes:

- El primer argumento permite definir el directorio donde se encuentra la aplicación, también define el nombre del archivo generado.
- El segundo argumento define la etiqueta asociada al carrusel de objetos generado.
- El tercer argumento define el número de versión de los módulos generados.
- El cuarto argumento permite definir el PID en el que se envía el carrusel de objetos, este valor pude ser desde 1 a 8191, es importante ya que se tiene que ingresar en las tablas PSI.
- El quinto argumento permite definir el identificador del carrusel de datos.

Para la generación del TS con una aplicación se usó el *script* con los siguientes argumentos: el primer argumento se especificó `yachay` que es la carpeta donde se encuentra la aplicación NCL con todos los elementos que intervienen en dicha aplicación; en el segundo argumento se colocó `0x0C`, el cual es un valor en hexadecimal que tienen todos los carruseles asociados y que se utilizarán en las tablas PSI; en el tercer argumentos se colocó `1`, el cual es la versión del carrusel de datos; en el cuarto argumento se especificó `2004`, que es el PID que se utilizará para las tablas PSI y en el último argumento se colocó el valor `2` el cual es el identificador del carrusel de datos en las tablas PSI.

```
oc-update.sh yachay 0x0C 1 2004 2
```

Código 3.15 Comando para la generación del TS que contiene la aplicación Ginga

Una vez obtenidos todos los TS a utilizar en el flujo único de paquetes de transporte TS es necesaria la creación de las tablas PSI puesto que sin esto no se podría

[8] Carrusel de objetos se utiliza para el envío de datos de manera cíclica que son utilizados en los sistemas de televisión digital.

interpretar correctamente el flujo único de paquetes de transporte TS. En el Anexo B-1 se presenta un archivo que está escrito en phyton para la creación de dichas tablas, este programa fue desarrollado por LIFIA, también se adjuntan en el Anexo B-2 el manual de uso para poder modificar los parámetros de las tablas PSI.

Una vez obtenidos todos los TS y las tablas PSI es momento de multiplexar todos estos elementos para la creación del flujo único de paquetes de transporte TS, esto se lo realiza con el comando `tscbrmuxer` que se presenta en el Código 3.16, a continuación se especifican los parámetros:

- El primer parámetro permite definir la cantidad de paquetes que se desean multiplexar.
- El segundo parámetro define la velocidad de transmisión de la tabla PAT y el archivo que lo contiene.
- El tercer parámetro permite definir la velocidad de transmisión de la tabla PMT y el archivo que lo contiene.
- El cuarto parámetro define la velocidad de transmisión de la tabla NIT y el archivo que lo contiene.
- El quinto parámetro especifica la velocidad de transmisión de la tabla SDT y el archivo que lo contiene.
- El sexto parámetro define la velocidad de transmisión de la tabla AIT con el archivo que lo contiene.
- El séptimo parámetro permite definir la velocidad de transmisión del TS que incluye la aplicación Ginga y el archivo que lo contiene.
- El octavo parámetro define la velocidad de transmisión del TS de video y el archivo que lo contiene.
- El noveno parámetro permite definir la velocidad de transmisión del archivo del TS de audio y el archivo del mismo.
- El décimo parámetro permite definir la velocidad de transmisión de los paquetes nulos y el archivo que lo contiene.

- Finalmente se define el nombre del archivo del flujo único de paquetes de transporte TS que se genera con su extensión respectiva.

Para generar el flujo único de paquetes de transporte TS se utilizó el comando `tscbrmuxer` que se presenta en el Código 3.16 el cual se conformó de la siguiente manera:

En la primera línea se colocó `7051326` que es el número total de paquetes que se desean multiplexar, este valor se lo obtuvo de la siguiente forma, el video de Yachay tiene una duración de cinco minutos con cincuenta y cuatro segundos, en la norma brasileña se transmiten 19.919 paquetes por segundo, multiplicando estos dos valores da como resultado 7'051.326 paquetes a multiplexar.

En la segunda línea se especificó `15040 pat.ts`, el primer valor se lo obtuvo de la siguiente forma, tanto la tabla PAT como la tabla PMT deben ser enviadas al menos 10 veces por segundo, estas tablas se transmiten en paquetes de 188 bytes, por lo tanto se requiere que la velocidad de transmisión sea de 15.040 bps y en el segundo valor se colocó el archivo que contiene la tabla PAT.

En la línea tercera se colocó `15040 pmt.ts`, el primer valor ya se mencionó como se lo obtuvo y el segundo valor es el archivo que contiene la tabla PMT.

En la cuarta línea se especifico `3008 nit.ts`, el primer valor se obtuvo de la siguiente manera, tanto la tabla NIT como la tabla SDT y la tabla AIT deben ser enviadas al menos 2 veces por segundo, el tamaño de estas tablas en el paquete es de 188 bytes, por lo que se requiere que la transmisión sea de 3.008 bps.

En la quinta línea se colocó `3008 sdt.ts`, el primer valor ya se mencionó como se lo obtuvo y el segundo valor es el archivo que contiene la tabla SDT.

En el sexto parámetro se especificó `3008 ait.ts`, el primer valor ya se lo mencionó y el segundo valor es el archivo que incluye la tabla AIT.

En la séptima línea colocó `357000 yachayncl.ts`, el primer valor es la velocidad de transmisión y el segundo valor específica el archivo TS que incluye la aplicación Ginga.

En la octava línea se especificó `4800000 yachayVideo.ts`, el primer valor específica la velocidad de transmisión que en este caso debe ser mayor que la tasa transmisión del video TS y también se agrega el archivo que contiene el archivo TS de video.

En la novena línea se colocó `198000 yachayAudio.ts`, el primer valor específica la velocidad de transmisión que en este caso debe ser mayor que la tasa transmisión del audio TS y se agrega el archivo que incluye el archivo TS de audio.

En el décima línea se especificó `22031190 null.ts`, el primer valor se obtuvo de la siguiente forma: la norma brasileña especifica una velocidad de transmisión fija de 29'958.294 bps, la velocidad de transmisión que se está generando es de 5'384.104 bps que resulta de la suma de todas las velocidades especificadas hasta este momento de cada uno de los elementos que conforman el flujo único de paquetes de transporte TS, entonces al restar el valor de la transmisión fija de la norma brasileña del valor antes mencionado se obtiene la velocidad de los paquetes nulos que es 22'031.190 bps y el segundo valor específica el archivo null que se creó junto a las tablas PSI, está formado por archivos nulos 0xFF con el PID 8191 y finalmente se guarda todo en el archivo generado con el nombre de `yachay.ts`.

Se generó el archivo que contiene el flujo único de paquetes de transporte TS, pero tiene un problema con el archivo generado con el PCR[9] que es la frecuencia del reloj del sistema que trabaja a 27 Mhz. Para solucionar este problema se utiliza el comando `tsstamp` definiendo la velocidad de transmisión de la norma brasileña como se presenta en el Código 3.17.

[9] PCR (*Program Clock Reference*) es una información de sincronización del reloj que se incluye periódicamente en los paquetes de transporte.

```
0       tscbrmuxer \
1       7051326 \
2       b:15040 pat.ts \
3       b:15040 pmt_sd.ts \
4       b:3008 nit.ts \
5       b:3008 sdt.ts \
6       b:3008 ait.ts \
7       b:357000 yachayncl.ts \
8       b:4800000 yachayVideo.ts \
9       b:198000 yachayAudio.ts \
10      b:22031190 null.ts > yachay.ts
```

Código 3.16 Generación de flujo único de paquetes de transporte TS

```
tsstamp yachay.ts 29958294 > yachay.fixed.ts
```

Código 3.17 Código reparar el flujo único de paquetes de transporte TS

En la Figura 3.35 se presenta una captura de pantalla de parte del video en el cual se observa el tema oferta académica de Yachay en particular se ha seleccionado la opción de ingeniería, en la parte inferior se puede observar las carreras que se ofertan, adicionalmente en la parte superior izquierda existe información adicional sobre las carreras como duración.

Figura 3.35 Captura de pantalla en la que se observa la oferta académica de Yachay

3.4 PRUEBAS CON EQUIPOS REALES

Inicialmente se hicieron pruebas en el simulador Ginga, las capturas se presentaron anteriormente. Las pruebas con equipos reales se presentan a continuación: en la Figura 3.36 se encuentran la imagen con equipos reales de la aplicación “Categorización de Universidades del Ecuador”; en la Figura 3.37 se presenta la imagen de la aplicación “Becas Universidades de Excelencia” con equipos reales; en la Figura 3.38 se muestran la imagen con equipos reales de la aplicación “SNNA” y en la Figura 3.39 se exhibe la imagen con equipos reales de la aplicación “Yachay ciudad del conocimiento”.

Para transmitir el flujo único de paquetes de transporte TS se empleó la aplicación SteamXpress, en la Figura 3.40 se presenta una captura de dicho software con la información de todas las velocidades de transmisión que se especificaron anteriormente. Cabe recalcar que los valores anteriormente mencionados se pueden apreciar, por ejemplo 3008 que es la velocidad de las tablas NIT y SDT están presentes en esta imagen.

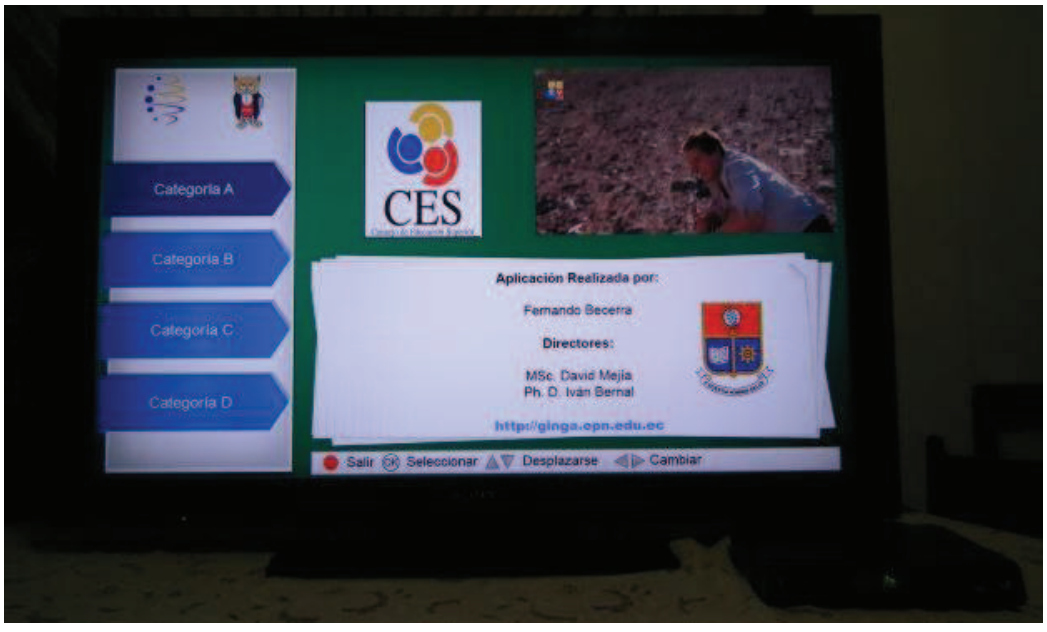

Figura 3.36 Pruebas con equipos reales de la aplicación “Categorización de Universidades del Ecuador”

Figura 3.37 Pruebas con equipos reales de la aplicación “Becas Universidades de Excelencia”

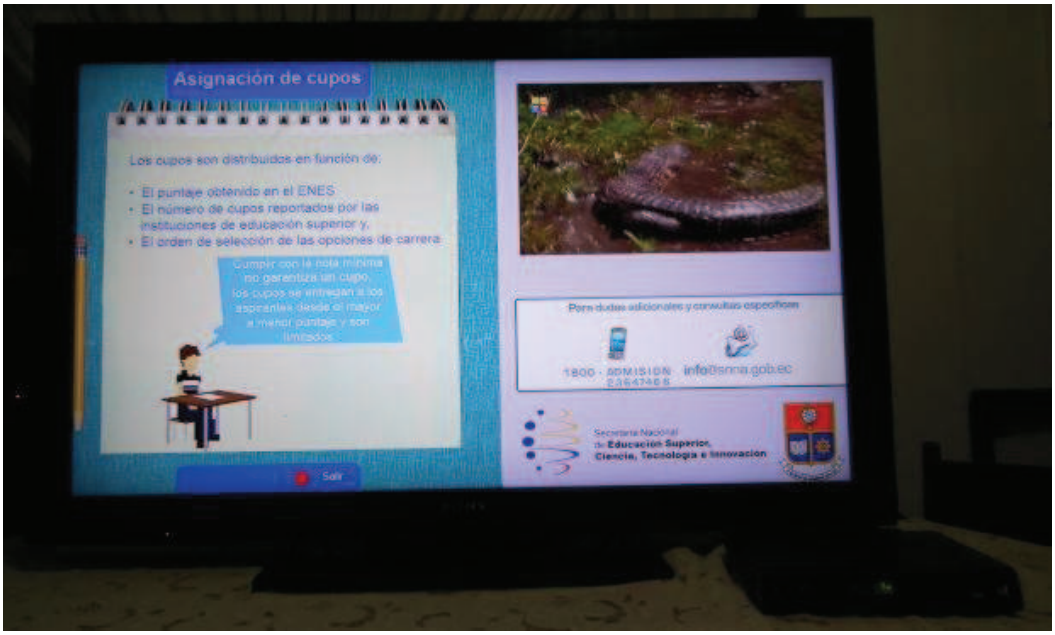

Figura 3.38 Pruebas con equipos reales de la aplicación “Sistema Nacional Nivelación y Admisión”

Figura 3.39 Pruebas con equipos reales de la aplicación “Ciudad del conocimiento Yachay”

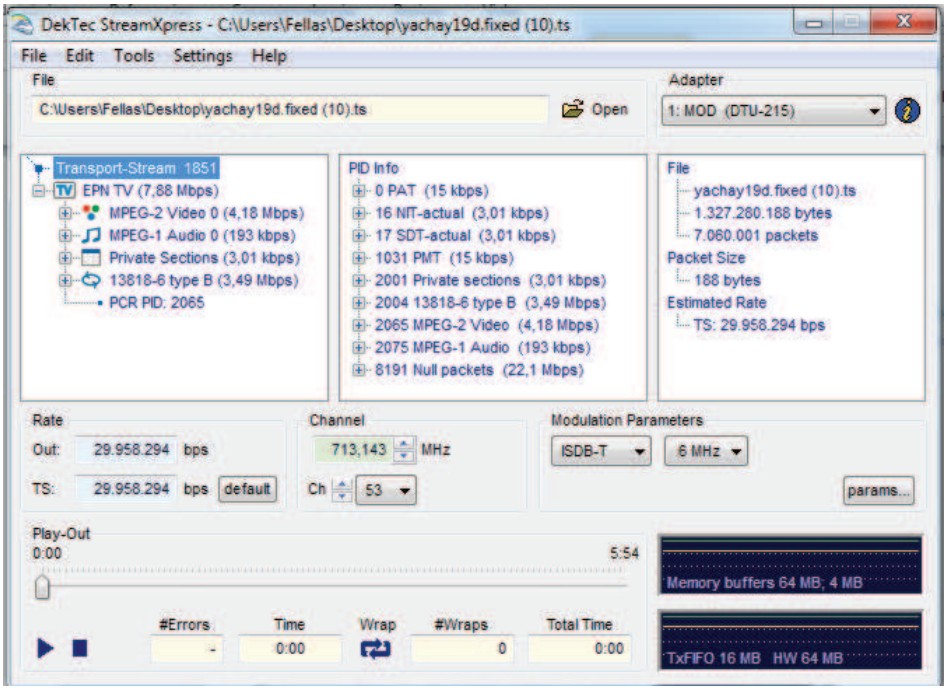

Figura 3.40 Software StreamXpress con los resultados del flujo único de paquetes de transporte TS

3.5 ENCUESTAS MOS [22], [23]

Las encuestas MOS, que en un inicio se utilizaron para VoIP[10], se emplean para conocer la percepción del usuario sobre el servicio de telefonía. Las encuestas MOS se expresan en números con cinco niveles de aceptación que van desde 1 que es el peor a 5 que es el excelente, estas encuestan varían entre los encuestados según su perspectiva. Hoy en día estas encuestas se ocupan para tener una percepción en el área de telecomunicaciones.

Cuando se obtiene un valor promedio de 4,0 a 4,5 en los resultados de las encuestas MOS se considera un servicio satisfactorio, valores menores a estos son inaceptables.

3.5.1 PARÁMETROS DE LAS ENCUESTAS MOS

Los parámetros que se tomaron en cuenta para las encuestas MOS se basaron en medir la percepción de los usuarios sobre las aplicaciones presentadas, para esto se emplearon los siguientes parámetros:

3.5.1.1 Diseño gráfico

Este parámetro proporciona el grado de aceptación de la interfaz gráfica por parte del usuario, hay que considerar que para este punto en particular, la información es vital debido a que no se empleó en el desarrollo de la aplicación la ayuda de un diseñador gráfico, por lo cual podría ser es el punto con mayores problemas.

3.5.1.2 Navegabilidad

Este parámetro permite calificar que tan sencillo fue la percepción de navegar dentro de la aplicación para el usuario con el uso del control remoto.

---

[10] VoIP(*Voice over* IP) llamado voz sobre IP es un grupo de recursos que hacen posible que la señal de voz viaje a través de la red empleando un protocolo IP.

3.5.1.3 Interactividad

Este parámetro permite medir la percepción del usuario respecto a la interactividad de la aplicación.

3.5.1.4 Contenido

Este parámetro es muy importante porque permite conocer si la percepción de la información que se presentó en la aplicación es la adecuada y le sirvió al usuario para mayor conocimiento de la temática tratada en las aplicaciones.

3.5.2 ENCUESTAS MOS DE LAS APLICACIONES DE TELEVISIÓN DIGITAL

Las encuestas MOS fueron realizadas a 50 usuarios finales. A todos los encuestados se les dio una explicación breve de lo que es la televisión digital y su rol con la interactividad, se permitió que el usuario manipule las aplicaciones e interactúe con las mismas. Cada aplicación Ginga tiene su propia encuesta para poder discriminar cada información propuesta. Las encuestas tuvieron las siguientes opciones para la calificación con su respectiva nota: malo (1), pobre (2), pasable (3), bueno (4) y excelente (5).

Cada una de las respuestas que se obtuvieron en las encuestas MOS se las proceso, obteniendo el valor de la media, desviación estándar y la mediana. Con estos valores se puede tener una noción de los resultados obtenidos como se explica a continuación: con la media se consigue el promedio de aceptación, con la desviación estándar se puede demostrar la semejanza de opiniones y con la mediana se obtiene un valor entero para poder encajar con las calificaciones especificadas.

3.5.2.1 Resultados de las encuestas MOS

Los resultados que se obtuvieron de las cuatro encuestas MOS se presentan a continuación:

En la Tabla 3.5 se presentan los resultados de la aplicación “Categorización de las Universidades del Ecuador”, en todas las preguntas se tiene una calificación buena, pero existen puntos relacionados con el diseño gráfico de la aplicación que son bajos a comparación de los otros resultados, por ejemplo el ícono de interactividad que es el ítem 1 el cual tuvo un media de 4.06. El punto más bajo de la calificación tiene que ver con el contenido de información que es el ítem 8, el cual tiene una media de 4 ya que esta aplicación redujo considerablemente el tamaño por los problemas antes mencionados, por esto solo se presentan ciertos aspectos de la categorización de las universidades. El ítem con más alta calificación es 6 con una media de 4.36 el cual está relacionado con la navegabilidad el cual a los usuarios les pareció adecuado.

Preguntas	**Media**	**σ**	**Mediana**
1) ¿Llama la atención el ícono de interactividad?	4,06	0,766	4
2) ¿El tamaño del texto presentado le pareció adecuado?	4,16	0,738	4
3) ¿En la información presenta, pudo navegar sin problema?	4,32	0,683	4
4) ¿En la información presentada, las imágenes le parecieron adecuadas?	4,14	0,880	4
5) ¿La ayuda para el uso del control remoto le pareció adecuada?	4,28	0,701	4
6) ¿La aplicación, le pareció fácil de utilizar?	4,34	0,592	4
7) ¿El contenido presentado, le pareció claro?	4,4	0,606	4
8) ¿En base de la información presentada, usted cree que sabe más sobre la categorización de las universidades?	4	0,638	4
9) ¿En base de la información presentada, se obtuvo una mejor visión de la oferta académica?	4,02	0,795	4
10) ¿Prestó mayor atención a la aplicación que a la programación normal de la televisión?	4,24	0,656	4
11) ¿Cree usted que la aplicación, cumple con el objetivo de informar claramente sobre la Categorización de las Universidades?	4,18	0,690	4
12) ¿De forma general que les pareció la aplicación, considerando las limitaciones que se ha tenido?	4,22	0,678	4
	4,196	0,702	4

Tabla 3.5 Encuesta MOS de la aplicación “Categorización de las Universidades de Ecuador”

Los resultados de la encuesta MOS de la aplicación “Becas Universidades de Excelencia” se presentan en la Tabla 3.6, de los cuales se puede observar que se obtuvo buena aceptación, además se aprecia claramente que la interactividad sobresale entre los demás ya que el ítem 7 tiene una media de 4,4 y una desviación estándar de 0.571 lo que demuestra la similitud de criterios de los usuarios, este ítem tiene buena calificación debido a que en esta aplicación se encuentra la información del *feed* RSS la cual presenta una información dinámica. También hay calificaciones bajas en la parte de diseño gráfico ya que el ítem 2 que se refiere al tamaño del texto tiene una media de 4.8 y una desviación estándar alta de 0.778 el cual representa que el tamaño del texto no es adecuado y por tanto se debería revisarlo.

Preguntas	**Media**	**σ**	**Mediana**
1) ¿Llama la atención el ícono de interactividad?	4,08	0,723	4
2) ¿El tamaño del texto presentado le pareció adecuado?	4,08	0,778	4
3) ¿En la información presenta, pudo navegar sin problema?	4,36	0,662	4
4) ¿La ayuda para el uso del control remoto le pareció adecuada?	4,16	0,738	4
5) ¿La aplicación, le pareció fácil de utilizar?	4,14	0,534	4
6) ¿Considera claro el contenido presentado?	4,22	0,678	4
7) ¿La interactividad presentada le pareció adecuada?	4,4	0,571	4
8) ¿EL uso de la interactividad ofrece información adicional sobre el tema tratado?	4,12	0,746	4
9) ¿La información desplegada sobre las becas, aclaró que requisitos debe reunir para aplicar a una de ellas?	4,2	0,755	4
10) ¿La presentación de las noticias en la aplicación dio un valor agregado a la información desplegada?	4,28	0,640	4
11) ¿Prestó mayor atención a la aplicación que a la programación normal de la televisión?	4,28	0,640	4
12) ¿Cree usted que la aplicación, cumple con el objetivo de informar cómo se puede acceder a las Becas de alto rendimiento?	4,32	0,683	4
13) ¿De forma general que les pareció la aplicación, considerando las limitaciones que se ha tenido?	4,34	0,658	4
	4,229	0,677	4

Tabla 3.6 Encuesta MOS aplicación de “Becas Universidades de Excelencia”

Los resultados de la encuesta MOS realizada a la aplicación “Sistema Nacional de Nivelación y Admisión” se pueden observar en la Tabla 3.7, se puede apreciar que los usuarios no están muy satisfechos con el ícono de interactividad y con el contenido sobre el SNNA que son los ítems 1 y 10 ya que en estos ítems tienen una media de 4.039 y 4,078 respectivamente, además que sus desviaciones estándar son muy altas ya que es superior a 0,7. Los otros ítems de la encuesta tuvieron una buena aceptación en términos generales, pero el ítem con una media más alta es el 3 con un valor de 4.33 y una desviación de 0,589, en este ítem podemos observar que la simplicidad del uso de la aplicación a los usuarios les pareció buena ya que la aplicación consta de un solo menú.

Preguntas	Media	σ	Mediana
1) ¿Llama la atención el ícono de interactividad?	4,039	0,799	4
2) ¿El tamaño del texto presentado le pareció adecuado?	4,176	0,654	4
3) ¿En la información presenta, pudo navegar sin problema?	4,333	0,589	4
4) ¿La ayuda para el uso del control remoto le pareció adecuada?	4,333	0,739	4
5) ¿La aplicación, le pareció fácil de utilizar?	4,294	0,610	4
6) ¿El contenido presentado le pareció claro?	4,294	0,642	4
7) ¿Prestó mayor atención a la aplicación que a la programación normal de la televisión?	4,255	0,659	4
8) ¿Con la información presentada, son claros los requisitos para rendir la prueba del SNNA?	4,137	0,849	4
9) ¿Con la información dada, son claros los pasos para la inscripción en la prueba del SNNA?	4,176	0,793	4
10) ¿Cree usted que la aplicación, cumple con el objetivo de informar sobre el SNNA?	4,078	0,688	4
11) ¿De forma general que les pareció la aplicación, considerando las limitaciones que se ha tenido?	4,333	0,653	4
	4,222	0,697	4

Tabla 3.7 Encuesta MOS de la aplicación “Sistema Nacional de Nivelación y Admisión”

Los resultados de la encuesta MOS de la aplicación “Ciudad del conocimiento Yachay” se pueden apreciar en la Tabla 3.8, las calificaciones son buenas ya que todos los ítems tienen una calificación mayor a cuatro, pero los ítems que menor calificación tuvieron fueron en la parte de contenidos como es el ítem 6 el cual tiene una media de 4,15 con una desviación estándar de 0.769 el cual es alto, los valores se obtuvieron porque la información presentada es pequeña porque se presenta en espacios muy cortos de tiempo, también la información puede desconcentrar al usuario de la programación normal de la televisión. El ítem 5 que se refiere a la utilización de la aplicación de parte del usuario tiene una media de 4.375 y una desviación estándar de 0.627, esta calificación es la más alta ya que la aplicación es sencilla de utilizar y en la mayoría de casos solo debe de presionar dos botones.

Preguntas	**Media**	**σ**	**Mediana**
1) ¿Llama la atención el ícono de interactividad?	4,25	0,630	4
2) ¿El tamaño del texto presentado le pareció adecuado?	4,35	0,579	4
3) ¿En la información presenta, pudo navegar sin problema?	4,2	0,822	4
4) ¿La ayuda para el uso del control remoto le pareció adecuada?	4,275	0,678	4
5) ¿La aplicación, le pareció fácil de utilizar?	4,375	0,627	4
6) ¿El contenido presentado le pareció claro?	4,15	0,769	4
7) ¿Prestó mayor atención a la aplicación que a la programación normal de la televisión?	4,15	0,699	4
8) ¿Le pareció adecuada la interactividad que se tuvo con el video y la información desarrollada?	4,2	0,723	4
9) ¿Con la Información desarrollada, el contenido de la oferta académica de la Universidad de Yachay le pareció adecuado?	4,25	0,707	4
10) ¿Con la Información desarrollada, sincronizada con el video le pareció que da un valor agregado a este?	4,175	0,712	4
11) ¿La aplicación, cumple con el objetivo de informar sobre la universidad de Yachay?	4,35	0,622	4
12) ¿De forma general que les pareció la aplicación, considerando las limitaciones que se ha tenido?	4,275	0,678	4
	4,25	0,687	4

Tabla 3.8 Encuesta MOS de la aplicación “Ciudad del conocimiento Yachay”

Inicialmente las interfaces de las aplicaciones eran diferentes y poco vistosas, por esta razón fueron mejorando en función de las pruebas que se realizaron con diferentes personas, de tal manera se puede apreciar que en las encuestas MOS realizadas la mayor parte de los encuestados coinciden que las aplicaciones interactivas son buenas. Con esto se puede concluir que las aplicaciones tienen una buena aceptación del público en general, ya que las encuestas fueron realizadas a diferentes tipos de personas como son: estudiantes universitarios, estudiantes de colegio, profesionales, entre otros la cual recoge diferentes tipos de opiniones.

REFERENCIAS BIBLIOGRÁFICAS

[1] Universidades y escuelas politécnicas: por categoría. *http://www.ces.gob.ec/ies/universidades-y-escuelas-politecnicas/por-categoria*. Consultado en Julio 2013.

[2] Really Simple Syndication. *http://www.ceaaces.gob.ec/index.php?option=com_content&view=article&id=19&Itemid=22*. Consultado en Julio del 2013

[3] Mandato Constituyente No.14. *EVALUACIÓN DE DESEMPEÑO INSTITUCIONAL DE LAS UNIVERSIDADES Y ESCUELAS POLITÉCNICAS DEL ECUADOR*. Fecha de edición 4 de noviembre de 2009.

[4] Becas Universidades de Excelencia. http://programasbecas.educacionsuperior.gob.ec/programas/becas-en-el-exterior/universidades-de-excelencia#descripci%C3%B3n. Consultado en Julio 2013.

[5] Sistema nacional de nivelación y admisión. *http://www.snna.gob.ec*. Consultado en Julio 2013.

[6] Ciudad del conocimiento Yachay. *http://www.yachay.ec*. Consultado en Julio 2013.

[7] Serrano Regol, Ivan. *Directrices para el Diseño de Interfaces de Televisión Digital Interactiva.* http://www.ivoserrano.com/ba/Directrices-dise%C3%B1o-interfaces-iTV.pdf. Consultado en Julio 2013.

[8] LEYES DE LA PERCEPCIÓN. *http://www.santiagoapostol.net/filosofia/percepcion.htm*. Consultado en Julio 2013.

[9] Lu, Karyn Y. INTERACTION DESIGN PRINCIPLES FOR INTERACTIVE TELEVISION. http://lmc.gatech.edu/ms_projects/klu/lu_karyn_y_200505_mast.pdf. Consultado en Julio 2013.

[10] Teoría del color. *http://www.fotonostra.com/grafico/teoriacolor.htm*. Consultado en Julio 2013.

[11] Composer NCL. *http://composer.telemidia.puc-rio.br*. Consultado en Julio 2013.

[12] NCL Eclipse. *http://laws.deinf.ufma.br/ncleclipse/pt-br:start?redirect=1#.UfgYhm1Z5vm*. Consultado en Julio 2013.

[13] GINGA NCL. *http://www.softwarepublico.gov.br/dotlrn/clubs/ginga/gingancl/xowiki/gingancl_vm*. Consultado en Julio 2013.

[14] Ginga4Windows released . *http://composer.telemidia.puc-rio.br/en/news/ginga4windows_0.13.1_released*. Consultado en Julio 2013.

[15] Adobe Photoshop. http://www.adobe.com/es/products/photoshop.html. Consultado en Julio 2013.

[16] Venegas Picón Luis Alberto. "GENERACIÓN DE UNA TRAMA BROADCAST TRANSPORT STREAM (BTS) USANDO EL SOFTWARE LIBRE OPENCASTER". Consultado en Julio 2013.

[17] MPEG-2 Transport Stream. *http://www.afterdawn.com/glossary/term.cfm/mpeg2_transport_stream*. Consultado en Julio 2013.

[18] OpenCaster. *http://www.avalpa.com/the-key-values/15-free-software/33-opencaster*. Consultado en Julio 2013.

[19] Avalpa. *Avalpa Broadcast Server User Manual*. http://www.avalpa.com/assets/freesoft/opencaster/AvalpaBroadcastServerUserManual-v2.1.pdf. Consultado en Julio 2013.

[20] Lifia. OpenCaster para SATVD-T. http://wiki.ginga.org.ar/lib/exe/fetch.php?media=lifia:guiaopencaster2.pdf. Consultado en Julio 2013.

[21] MPEG 2 . http://wikitel.info/wiki/MPEG_2. Consultado en Julio 2013.

[22] MOS- Mean Opinion Score for VoIP Testing. http://www.voipmechanic.com/mos-mean-opinion-score.htm. Consultado en Julio 2013.

[23] Mean Opinion Score (MOS). http://voip.about.com/od/voipbasics/a/MOS.htm. Consultado en Julio 2013.

CAPÍTULO 4

CONCLUSIONES Y RECOMENDACIONES

4.1 CONCLUSIONES

- Se concluye en base a las encuestas MOS que al usuario no le agrada mucho la parte visual, esta parte se puede trabajar. Es claro que existe problemas en la parte visual ya que no se tiene los conocimientos en diseño gráfico por lo cual es importante para diseñar estas aplicaciones que en el equipo de trabajo exista una persona que conozca del tema.
- Una vez analizados los resultados de las encuestas MOS la aplicación que presenta una mejor aceptación por parte de loa usuarios es la que exhibía la información de un *feed* RSS, debido a que los usuarios esperan tener información que venga desde el Internet, no únicamente información estática porque le va a parecer repetitiva.
- El software OpenCaster tiene limitaciones dado que la empresa que realizó este software no ha hecho cambios desde el 2011, además no tiene soporte para ISDB-tb y como se mencionó anteriormente el software utilizado esta parchado. Las ventajas de utilizar este software son dos: es gratuita y de código abierto. Se concluye que consultando con el manual de AVALPA se puede corregir los problemas que se puedan encontrar al realizar una trama TS.
- El uso del software OpenCaster está en función de los parámetros, se debe conocer cómo funcionan para poder determinar que parámetros se tienen que especificar y en función de éstos poder generar la trama de forma adecuada, es importante saber cuál es la velocidad de transmisión de cada uno de los elementos porque en función de eso se tendrán que poner los paquetes nulos para que la trama TS se genere correctamente.

- En la creación del flujo único de paquetes de transporte TS se deben tomar en cuenta muchos aspectos que pueden hacer fallar la sincronización de la aplicación con el video. Es necesario tomar en cuenta el tiempo que se demora el STB en recibir la aplicación porque a lo que termina de recibir la aplicación recién inicia a procesarla y corre el tiempo de inicio de la aplicación.
- La utilización de los colores en las aplicaciones realizadas fueron muy importantes dado que un color le da vida a la aplicación y se presenta de forma correcta y clara ante el usuario final. Con esto se puede concluir que si las aplicaciones diseñadas no tienen colores atractivos en la presentación al usuario no le va a interesar y no utilizará la aplicación.
- Es importante considerar que las aplicaciones desarrolladas para Ginga no deben pesar más de 6 MB ya que si se sobrepasan este valor los STB van a tener problema dado que no tienen una memoria muy grande para almacenar las aplicaciones, si se carga una aplicación que pese más de 6 MB se corre el riesgo de que algunas imágenes no se presenten. También si las aplicaciones son muy grandes éstas tomaran mucho tiempo para almacenarse dentro del STB y de pronto el usuario se aburre y puede cambiar de programa o de canal.
- Al utilizar el lenguaje de programación Lua se brinda una interactividad superior a las aplicaciones que normalmente se han diseñado, además con la utilización del canal de retorno se puede obtener información adicional como se lo realizó con los *feed* RSS proporcionando un valor agregado a las aplicaciones puesto que se obtiene una aplicación dinámica y cada vez que el usuario la utilice se presentará información nueva, de esta manera el usuario no percibirá una aplicación aburrida.
- El uso del software *Composer* NCL también brinda una ayuda en el diseño gráfico de las áreas que contengan imágenes porque ofrece una percepción espacial de la ubicación que va a obtener una imagen dentro de una aplicación además del tamaño, con esto es más sencillo a la hora de diseñar

la aplicación ya que las imágenes que se van a utilizar estarán bien dimensionadas y cuando se presenten no se van a distorsionar, con esto se obtiene una aplicación visualmente adecuada sin problemas en las imágenes.

- Para ingresar textos informativos deben se ser lo más precisos posibles porque en la interfaz de la televisión no se puede usar mucho espacio o sobrecargar de información ya que los usuarios cuando están viendo la televisión la usan para relajarse, no la usan como un medio de información. Si la información es muy extensa se la debe resumir o agregar un *link* para que si los usuarios están interesados en ese tema empleen el *link* y se informen de una manera más completa.
- El *plugin* que se desarrolló permite que personas que no tienen experiencia sobre los lenguajes de programación NCL y Lua puedan crear sus propias aplicaciones para televisión digital, que obtengan información desde un *feed* RSS y poderlas presentar en pantalla.
- Como la televisión es un medio de comunicación masivo con gran aceptación en la sociedad ecuatoriana, las aplicaciones realizadas podrán ser vistas por un gran número de personas, permitiendo difundir el tema de educación superior a nivel masivo.

4.2 RECOMENDACIONES

- Para el desarrollo ágil de las aplicaciones se recomienda que no solo una persona sea responsable de su desarrollo, sino que sean varias, es decir que sea una persona la encargada de la parte técnica o de programación, otra un diseñador de la interfaz gráfica y una persona encargada de contenidos. Con esto se tendría un equipo de trabajo completo en los tres temas fundamentales para que las aplicaciones desarrolladas sean del agrado del usuario y que estén satisfechos con toda la información expuesta.
- Para utilizar el canal de retorno se recomienda tener el dispositivo configurado de forma correcta en la parte de red porque el STB de la marca EiTV tiene un problema con el DNS automático, por lo cual se recomienda que para su uso

siempre se especifique un DNS, como por ejemplo el de Google cuya dirección IP puede ser 8.8.8.8 o 4.4.4.4.

- Además de probar las aplicaciones en el STB virtual es importante que sea probado en equipos reales ya que al momento que se presentan las aplicaciones en estos equipos se pueden presentar errores visuales que en los simuladores no se los detectaron, debido a que usualmente el Ginga implementado en el simulador es más completo que el que se incluye en el STB real.
- En particular cuando se utiliza el STB de la marca EiTV se observa que los caracteres especiales del idioma español como son las tildes o la “ñ” no son presentados de forma adecuada porque trata de presentar caracteres propios del portugués, por lo cual se recomienda que mientras no se tenga un soporte completo del lenguaje español se prefiera el uso de imágenes en vez de texto a pesar de que las imágenes ocupan más de espacio en memoria, pero se evita que se presenten caracteres extraños en vez de los caracteres propios del español.
- Se recomienda que los títulos del menú sean concisos y que no sean muy extensos porque entre más grande sea el título del ítem el televidente tendrá que analizarlo y posiblemente no le gustará la aplicación ya que los televidentes usan la televisión para relajarse. Si los títulos de los ítems de un menú utilizados en una aplicación no son claros y concisos el televidente va a tener una idea equivocada de la información presentada en cada ítem, resultado de esto el televidente no va a encontrar la información deseada y le parecerá que la aplicación desarrollada no es útil.
- Se intentó utilizar en las aplicaciones un carrusel de imágenes el mismo que en el simulador funcionaba correctamente, pero cuando se utilizó en equipos reales como se puede observar en la Figura 4.1 se obtuvieron varios errores de sobre posición de imágenes, dando un mal aspecto a la aplicación desarrollada, por esta razón se recomienda no utilizar un carrusel de imágenes en este tipo de aplicaciones, además es difícil obtener un tiempo

medio para el intercambio de imágenes porque el tiempo que un usuario lee el texto expuesto es variable, por esta razón es mejor utilizar los cursores para cambiar las imágenes de la aplicación así los usuarios podrán cambiarlas a su criterio.

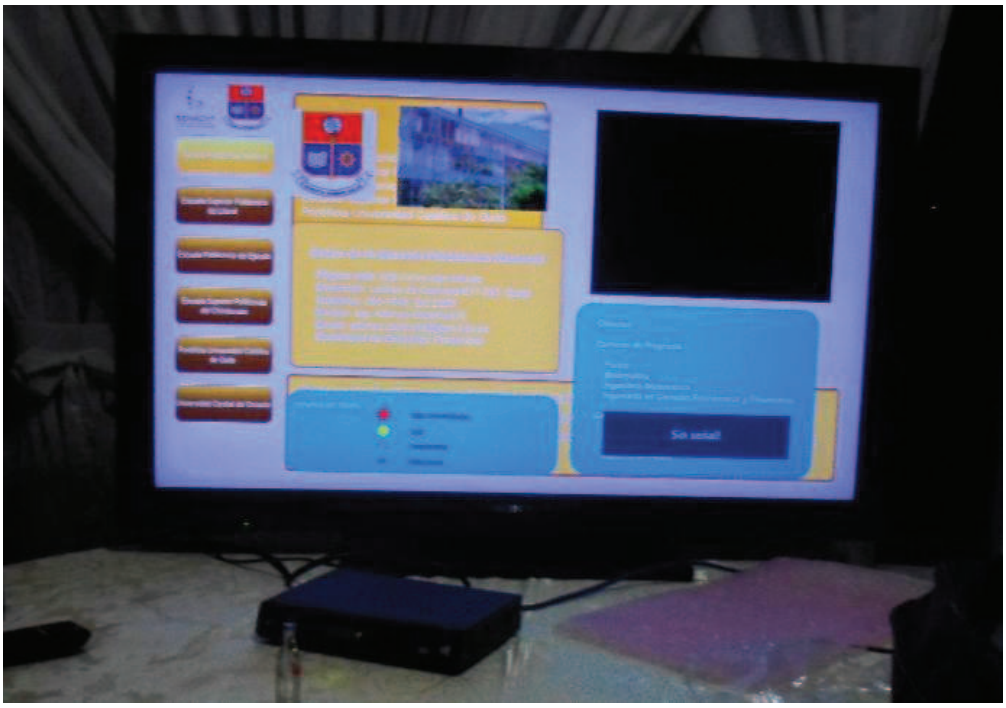

Figura 4.1 Error de sobre posición de imágenes en el STB de EiTV

REFERENCIAS BIBLIOGRÁFICAS

[1] ABNT NBR 15606-2, *http://www.abnt.org.br/imagens/Normalizacao_TV_Digital/ABNTNBR15606-2_2007Ing_2008.pdf*. Consultado en enero 2013

[2] Adobe Photoshop. http://www.adobe.com/es/products/photoshop.html. Consultado en Julio 2013.

[3] Anonimo . *"APRENDA Qt4 DESDE HOY MISMO"*. http://www.choccac.com/attachments/article/11/Aprenda_Qt4_Hoy_Mismo.pdf. Consultado en marzo de 2013.

[4] ATSC, DTMB, DVB-T/DVB-T2 E ISDB-T. *http://es.dtvstatus.net/*. Consultado en Eenero 2013.

[5] Avalpa. *Avalpa Broadcast Server User Manual*. http://www.avalpa.com/assets/freesoft/opencaster/AvalpaBroadcastServerUserManual-v2.1.pdf. Consultado en Julio 2013.

[6] Becas Universidades de Excelencia. http://programasbecas.educacionsuperior.gob.ec/programas/becas-en-el-exterior/universidades-de-excelencia#descripci%C3%B3n. Consultado en Julio 2013.

[7] Benoit, Herve. *Digital Television: Satellite, Cable, Terrestrial, IPTV, Mobile TV in the DVB Framework*. 2008.

[8] Blanchette, Jasmin. *"C++ GUI Programming with Qt 4"*. http://grimaldi.univ-tln.fr/Qt/C++-GUI-Programming-with-Qt-4-1st-ed.pdf. Consultado en marzo de 2013.

[9] Ciudad del conocimiento Yachay. *http://www.yachay.ec*. Consultado en Julio 2013.

[10] Composer NCL. *http://composer.telemidia.puc-rio.br*. Consultado en Julio 2013.

[11] COMUNIDAD GINGA ECUADOR. *http://comunidadgingaec.blogspot.com/search/label/¿Qué es Ginga%3F*. Consultado en Enero del 2013.

[12] Cubero, Manuel. *La Televisión Digital: Fundamentos y teorías*. 2009.

[13] DESARROLLO DE KANBAN. *http://www.desarrolloweb.com/articulos/desarrollo-agil-kanban.html*. Consultado en Febrero 2013.

[14] Eliomark, Pinzón Rondón. "*AGGREGATORS"*. http://www.slideshare.net/eliomark/diapositivas-rss-eliomark. Consultado en Marzo 2013.

[15] Exemplo 6 - Consulta ao Google. *http://www.telemidia.puc-rio.br/~francisco/nclua/tutorial/exemplo_06.html*. Consultado en Marzo 2013.

[16] Filho, Guido Lemos de Souza. *Ginga-J: The Procedural Middleware for the Brazilian Digital TV System*. 2005.

[17] FORO OFICIAL DE SBTVD. *http://forumsbtvd.org.br*. Consultado en Enero del 2013.

[18] GENERACIÓN DE UNA TRAMA BROADCAST TRANSPORT STREAM (BTS) USANDO EL SOFTWARE LIBRE OPENCASTER". Consultado en Julio 2013.

[19] GINGA NCL. *http://www.softwarepublico.gov.br/dotlrn/clubs/ginga/gingancl/xowiki/gingancl_vm*. Consultado en Julio 2013.

[20] Ginga4Windows released . *http://composer.telemidia.puc-rio.br/en/news/ginga4windows_0.13.1_released*. Consultado en Julio 2013.

[21] Henrik Kniberg & Mattias Skarin . *"Kanban y Scrum – obteniendo lo mejor de ambos".* http://www.proyectalis.com/documentos/KanbanVsScrum_Castellano_FINAL-printed.pdf. Consultado en Febrero 2013.

[22] How to: Create an NCL Composer Plugin . http://composer.telemidia.puc-rio.br/en/doc/tutorial/how_to_create_a_plugin_to_ncl_composer. Consultado en Julio 2013

[23] Icai, Pablo J. "Really Simple Syndication (RSS)". http://es.scribd.com/doc/56392931/Sistemas-Distribuidos-RSS. Consultado en Marzo 2013.

[24] Ierusalimschy, Roberto. *Programming in Lua*. 2006.

[25] Introduction to RSS. *http://www.webreference.com/authoring/languages/xml/rss/intro/index.html*. Consultado en Marzo 2013.

[26] Lekakos, George. *Interactive Digital Television: Technologies and Applications*. 2008.

[27] LEYES DE LA PERCEPCIÓN. *http://www.santiagoapostol.net/filosofia/percepcion.htm*. Consultado en Julio 2013.

[28] Lifia. OpenCaster para SATVD-T. http://wiki.ginga.org.ar/lib/exe/fetch.php?media=lifia:guiaopencaster2.pdf. Consultado en Julio 2013.

[29] Lu, Karyn Y. INTERACTION DESIGN PRINCIPLES FOR INTERACTIVE TELEVISION. http://lmc.gatech.edu/ms_projects/klu/lu_karyn_y_200505_mast.pdf. Consultado en Julio 2013.

[30] Mandato Constituyente No.14. *EVALUACIÓN DE DESEMPEÑO INSTITUCIONAL DE LAS UNIVERSIDADES Y ESCUELAS POLITÉCNICAS DEL ECUADOR*. Fecha de edición 4 de noviembre de 2009.

[31] Marcelo Hernán, Schenone. "*Diseño de una Metodología Ágil de Desarrollo de Software"*. http://materias.fi.uba.ar/7500/schenone-tesisdegradoingenieriainformatica.pdf. Consultado en Enero del 2013.

[32] Mean Opinion Score (MOS). http://voip.about.com/od/voipbasics/a/MOS.htm. Consultado en Julio 2013.

[33] Módulo canvas. *http://www.telemidia.puc-rio.br/~francisco/nclua/referencia/canvas.html*. Consultado en marzo de 2013.

[34] Moreira Gomes, Johnny. *"Utilizando o Lua Xml Parser para leitura simples de arquivos xml em aplicações NCL/Lua"*. http://www.ufjf.br/lapic/files/2010/06/Tutorial_Lua_XML_Parser.pdf. Consultado en marzo de 2013.

[35] MOS- Mean Opinion Score for VoIP Testing. http://www.voipmechanic.com/mos-mean-opinion-score.htm. Consultado en Julio 2013.

[36] MPEG 2 . http://wikitel.info/wiki/MPEG_2. Consultado en Julio 2013.

[37] MPEG-2 Transport Stream. *http://www.afterdawn.com/glossary/term.cfm/mpeg2_transport_stream*. Consultado en Julio 2013.

[38] NCL Eclipse. *http://laws.deinf.ufma.br/ncleclipse/pt-br:start?redirect=1#.UfgYhm1Z5vm*. Consultado en Julio 2013.

[39] OpenCaster. *http://www.avalpa.com/the-key-values/15-free-software/33-opencaster*. Consultado en Julio 2013.

[40] PÁGINA OFICIAL DE ATSC. *http://www.atsc.org/cms/index.php/standards/standards?layout=default*. Consultado en Enero 2013.

[41] PÁGINA OFICIAL DE COMPOSER NCL. *http://composer.telemidia.puc-rio.br*. Consultado en Enero 2013.

[42] PÁGINA OFICIAL DE DVB. *http://www.dvb.org/*. enero 2013.

[43] PÁGINA OFICIAL DE ISDB-T. *http://www.dibeg.org/techp/techp.html*. Consultado en Enero 2013.

[44] Pagina oficial de Qt Creator. http://qt-project.org/wiki/category:tools::qtcreator. Consultado en Mayo 2013.

[45] Paucar Curasma, Ronald. *"Análisis y Modelamiento de las Técnicas de Canal de Retorno e Interactividad para el Estándar de Televisión Digital Terrestre ISDB-T"*. publicado en 10 de Diciembre de 2010

[46] Penades, Maria Carmen. *"Metodologias Aguiles en el desarrollo de Software"*. http://www.willydev.net/descargas/masyxp.pdf. Consultado en Febrero 2013.

[47] Qué es el método Kanban para la gestión de proyectos?. *http://www.javiergarzas.com/2011/11/kanban.html*. Consultado en Febrero 2013.

[48] Really Simple Syndication. *http://www.ceaaces.gob.ec/index.php?option=com_content&view=article&id=19&Itemid=22*. Consultado en Julio del 2013

[49] Really Simple Syndication. *http://www.reallysimplesyndication.com/*. Consultado en Marzo del 2013.

[50] RSS 2.0 at Harvard Law. *http://cyber.law.harvard.edu/rss/rss.html*. Consultado en Marzo 2013.

[51] Serrano Regol, Ivan. *Directrices para el Diseño de Interfaces de Televisión Digital Interactiva.* Consultado en Julio 2013.

[52] Serrano Regol, Ivan. *Directrices para el Diseño de Interfaces de Televisión Digital Interactiva.* http://www.ivoserrano.com/ba/Directrices-dise%C3%B1o-interfaces-iTV.pdf. Consultado en Julio 2013.

[53] Sistema nacional de nivelación y admisión. *http://www.snna.gob.ec*. Consultado en Julio 2013.

[54] Soares, Luiz Fernando Gomes. *Progrmando em NCL 3.0*. 2012.

[55] SUPERTEL. *INFORME PARA LA DEFINICIÓN E IMPLEMENTACIÓN DE LA TELEVISIÓN DIGITAL TERRESTRE EN EL ECUADOR*. 2010.

[56] TELEVISIÓN DE ALTA DEFINICIÓN. *http://wikitel.info/wiki/Televisi%C3%B3n_de_alta_definici%C3%B3n*. Consultado en Enero 2013.

[57] TELEVISIÓN DIGITAL - VENTAJAS Y DESVENTAJAS. *http://observatoriotecedu.uned.ac.cr/index.php/actualidad/soportes/106-television-digital.html?start=2*. Consultado en Enero 2013.

[58] TELEVISIÓN DIGITAL. *http://www.fayerwayer.com/2010/10/desarrollan-el-primer-receptor-europeo-de-television-en-movimiento.* Consultado en Enero 2013.

[59] Teoría del color. *http://www.fotonostra.com/grafico/teoriacolor.htm*. Consultado en Julio 2013.

[60] Universidades y escuelas politécnicas: por categoría. *http://www.ces.gob.ec/ies/universidades-y-escuelas-politecnicas/por-categoria*. Consultado en Julio 2013.

[61] Varios autores. "*HD Magazine Nro 5"*. http://www.hdmagazine.org/. Consultado en Enero del 2013.

[62] Venegas Picón Luis Alberto.

[63] VENTAJAS FRENTE A LA TV ANALÓGICA. *http://www.canariastdt.es/index.php?option=com_content&view=article&id=21&Itemid=15*. Consultado en Enero 2013.

[64] What Is RSS. *http://www.xml.com/pub/a/2002/12/18/dive-into-xml.html.* Consultado en Febrero 2013.

[65] What Is RSS? RSS Explained. *http://www.whatisrss.com/*. Consultado en Enero del 2013.

[66] What is RSS?. *http://www.hardware-revolution.com/resources/rss-2/*. Consultado en Marzo 2013.

ANEXOS

ANEXO A

Código de las aplicaciones desarrolladas y del *plugin*

- A-1: Aplicación “Categorización de las universidades del Ecuador”
- A-2: Aplicación “Becas Universidades de Excelencia”
- A-3: Aplicación “Sistema nacional de Nivelación y Admisión”
- A-4: Aplicación “Yachay ciudad del Futuro”
- A-5: Código *plugin* consumidorRSS

(CD Adjunto)

ANEXO B

Código de las tablas PSI y manual AVALPA

- B-1: Código en phyton para la creación de las tablas PSI
- B-2: Manual de AVALPA

(CD Adjunto)