QuizTV: A Game for Interactive Digital Television. Development Considerations Using the Ginga-NCL Middleware.

Xavier Garnica-Bautista[1], Xavier Maita-Tepán[2], Magali Mejía-Pesántez[3], Lissete Muñoz-Guillén[4]
Departamento de Ciencias de la Computación
Universidad de Cuenca
Cuenca, Ecuador
{xavier.garnica[1], xavier.maita[2], magali.mejia[3], lissette.munozg[4]}@ucuenca.edu.ec

***Resumen*—This paper presents the process of analysis, design and implementation of a game for Interactive Digital Television (iDTV), using the Ginga middleware with NCL and Lua as programming languages. It also describes the game's functionality, its architecture and the methodology used for its development. In addition, it defines the necessary tools for the game's implementation, the considerations and limitations found especially in the design of graphical interfaces and the development as such of the application. As a final result, a visually attractive and interactive game was obtained, which makes use of a return channel to connect to a server and a database.**

***Keywords*—Interactive digital television, Ginga, NCL, Lua, ISDB-Tb.**

I. INTRODUCCIÓN

La Televisión Digital (TVD) ha traído consigo grandes beneficios en cuanto a rendimiento con respecto a la transmisión de televisión analógica, mejorando la calidad tanto de imagen como de sonido. En la TVD, utilizando el mismo ancho de banda, se pueden transmitir diferentes tipos de contenidos: programas de televisión de alta definición (HDTV), programas de definición estándar (SDTV), programas de audio y contenido multimedia entre otros [1]. Sin embargo, una de las principales características radica en la posibilidad de incluir datos para aplicaciones que no necesariamente tienen que ver con la programación emitida, por ejemplo juegos, mejorando así la experiencia del televidente [2].

El concepto de la Televisión Digital Interactiva (TVDi), surge cuando se permite la participación activa del televidente con el contenido emitido, permitiéndole determinar la dirección del flujo de dicho contenido. Es decir, el televidente es capaz de interactuar con la TV pasando de ser un simple espectador a ser un generador de información [3]. A su vez, ha permitido el despliegue de una gran cantidad de aplicaciones de diferentes tipos: t-learning [2] [4], t-commerce [5], juegos [6] [7], etc., en donde el televidente es el principal protagonista.

Varios son los estándares para la Televisión Digital Terrestre (TDT) adoptados por diferentes países. El presente trabajo se centra en el estándar brasileño llamado *International System for Digital Television* (ISDB-Tb), el cual está basado en el estándar japonés denominado *Integrated Services Digital Broadcasting - Terrestrial* (ISDB-T) [8]. ISDB-Tb permite la transmisión digital de videos tanto estándar como de alta definición, transmisión para dispositivos fijos, móviles y portátiles, e interactividad; además, propone el uso de un *Set-Top-Box* (STB) que ejecute el middleware *Ginga* diseñado específicamente para este estándar [9].

Ginga es el nombre del middleware de la recomendación ITU-T para servicios de IPTV y del Sistema Brasileño de TV Digital Terrestre, especificado con el objetivo de abstraer la plataforma de hardware para los desarrolladores [10]; posee dos entornos para facilitar el desarrollo de aplicaciones:

- *Ginga-J*: un entorno basado en Java para la ejecución y desarrollo de aplicaciones interactivas de tipo procedimental [11].
- *Ginga-NCL*: un entorno para aplicaciones declarativas basadas en el Lenguaje de Contexto Anidado (NCL, por sus siglas en inglés *Nested Context Language*) [12], el cual puede ser apoyado por scripts del lenguaje Lua para ampliar las funcionalidades de las aplicaciones interactivas.

En este contexto, desde que se adoptó el estándar *ISDB-Tb* para la TVD en Ecuador en el año 2010 [13], se ha abierto un nuevo campo de investigación en el país. Varios proyectos, aplicaciones y diversos estudios basados en el middleware *Ginga* han sido publicados, la mayoría no hacen énfasis en aspectos de programación y de uso de las herramientas disponibles para el desarrollo [14], generalmente se enfocan en medir el rendimiento del flujo de transmisión [15] [16] o presentar arquitecturas de desarrollo utilizadas [17]. Es por ello que este trabajo se enfoca en exponer el proceso de análisis, diseño e implementación de una aplicación para la TVDi, presentar las herramientas usadas y sus limitaciones, problemas surgidos durante el desarrollo y las soluciones implementadas. El objetivo es dar a conocer la experiencia adquirida durante el desarrollo, usando el middleware *Ginga*, empleando los lenguajes *NCL* y *Lua*, un canal de retorno, un servidor y una base de datos.

Lo que sigue de este artículo esta organizado de la siguiente manera: La sección II presenta los trabajos relacionados, es decir, aquellos que exponen procesos de implementación de juegos desarrollados para el entorno de la TVDi, usando el middleware *Ginga*. La sección III detalla las funcionalidades

978-1-5386-7612-7/18/$31.00 ©2018 IEEE
DOI 10.1109/INCISCOS.2018.00043

de la aplicación y la arquitectura planteada para su funcionamiento. La sección IV expone el proceso de desarrollo, utilizando como base la metodología Ágil TVDi. La sección V expone los resultados obtenidos, la sección VI presenta las dificultades encontradas. Finalmente, en la sección VII se presentan las conclusiones y trabajos futuros.

II. TRABAJOS RELACIONADOS

La experiencia televisiva ha ido mejorando y cambiando paulatinamente, permitiendo hoy en día, la participacion activa del televidente. En esta sección se hace una revisión de trabajos que presentan procesos de implementación de juegos interactivos, desarrollados para el entorno de la TVD, usando el middleware *Ginga* del estándar brasileño *ISDB-Tb*. Estas iniciativas permiten conocer las diferentes arquitecturas y desafios en el desarrollo así como los avances a lo largo del tiempo, tanto en la parte visual como interactiva.

En [6], se presenta el juego interactivo denominado *RummiTV*, que consiste en un tablero con varias fichas; el objetivo principal es que el jugador se deshaga de las fichas que le fueron asignadas, para esto, el jugador debe formar grupos de piezas y realizar una serie de operaciones con ellas (por color o números) (Fig. 1). Para el desarrollo del juego se aplicaron los enfoques procedimental y declarativo, obteniendo una aplicación híbrida para el entorno de la TVD. El enfoque procedimental consistió en el desarrollo de la lógica del juego usando el lenguaje de programación *Java*, y el enfoque declarativo fue utilizado para crear un manual de instrucciones que incluye toda la documentación sobre cómo jugar y la navegación dentro de la interfaz de usuario, mediante el lenguaje *NCL*. Aunque ambos enfoques fueron empleados en el desarrollo, los autores indican que las partes procedimentales y declarativas aún no son totalmente integradas por el middleware.

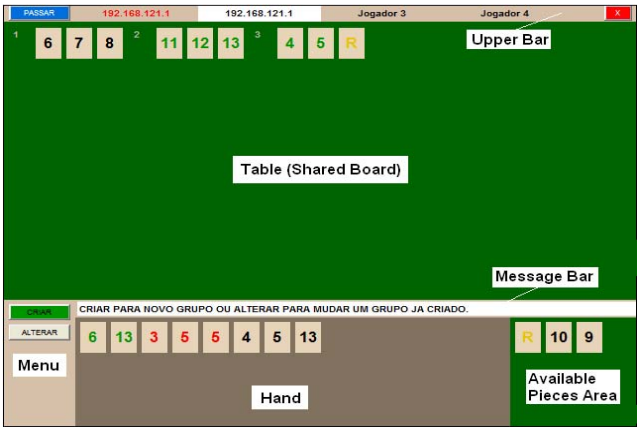

Fig. 1. Tablero y fichas del juego de RummiTv. Fuente: [6].

En [18], se describe el proceso de creación de juegos interactivos para el middleware *Ginga* usando los lenguajes *NCL* y *Lua*. Además, se propone una metodología la cual está conformada por cuatro etapas que consisten en: análisis de funciones proporcionadas por el middleware, la especificación del juego, la definición de las herramientas y la implementación. En el trabajo se presentan también algunas consideraciones y limitantes de las herramientas usadas. Los autores mencionan que se produjeron seis juegos para validar el proceso, uno de ellos es un juego de estrategia *Wing Force* (Fig. 2), el cual tiene como propósito conseguir que el jugador destruya aeronaves enemigas, tanques, barcos y elementos interactivos del escenario del juego en cada etapa.

Fig. 2. Interfaz gráfica del juego Wing Force. Fuente: [18].

Se han publicado varios marcos de desarrollo de juegos interactivos para la TVD, que tienen como objetivo facilitar y simplificar la implementación de dichas aplicaciones que funcionan bajo el middleware *Ginga*. Uno de estos marcos de desarrollo se presenta en [19], el cual actúa como una interfaz de programación de aplicaciones (API) de alto nivel para los desarrolladores de *Ginga-J* y como módulos para los de *Ginga-NCL*. Debido a este marco de desarrollo, los programadores no requieren conocimiento del lenguaje *NCL*, únicamente es necesario el conocimiento de *Lua* considerado un lenguaje más familiar. Una de las aplicaciones implementadas usando este marco de desarrollo es *GingaHero* (Fig. 3), un juego musical en el cual el televidente debe presionar los botones del control remoto según el tiempo exacto en el que las notas musicales se desplazan en la pantalla de juego.

Fig. 3. Interfaz gráfica del juego *GingaHero*. Fuente: [19].

Estos trabajos han permitido apreciar un panorama general de aplicaciones con el middleware *Ginga* utilizando lenguajes como *NCL* y *Lua*, pero en ninguno de estos trabajos expone como se llevo a cabo dichas aplicaciones en cuanto al desarrollo, herramientas, codificación, arquitectura y como se gestiono los componentes gráficos siendo estas experiencias de gran utilidad para el diseño y desarrollo de la aplicación que se presenta la siguiente sección de este trabajo.

III. Análisis y diseño de la aplicación

En esta sección se describe cada una de las funcionalidades de la aplicación y se presenta la arquitectura empleada para el desarrollo de cada uno de sus componentes.

A. Funcionalidad

QuizTV es un juego para la TVDi, que le permite al televidente responder trivias referentes al contenido que se muestra en los programas transmitidos por un canal de TV específico. Cada canal de TV tiene la posibilidad de definir una serie de preguntas con sus posibles respuestas en un cuestionario precargado. El televidente, luego de registrarse en el juego, podrá elegir qué trivia o cuestionario desea responder en cualquier instante, de la lista disponible. A continuación, se describen las funcionalidades de la aplicación:

1) Registro de usuarios: El televidente, al acceder a la aplicación, debe registrarse previamente, proporcionando información personal básica como: nombre de usuario, correo electrónico, contraseña, lugar y país de origen, entre otros. Además, tienen la posibilidad de seleccionar un avatar como imagen de perfil.

2) Inicio de sesión: Una vez registrados, los televidentes pueden acceder a la aplicación con su nombre de usuario y contraseña.

3) Perfil de usuario: El televidente puede editar su información de perfil, modificar su avatar o contraseña en cualquier momento.

4) Selección de parámetros de juego: El juego cuenta con una lista de categorías (series, películas, deportes, reality shows, etc.), el televidente selecciona una categoría y dentro de esta, el programa de TV del cual desea responder las trivias, finalmente, selecciona el modo de juego "contra reloj" o "contra otro jugador" con el que deberá responder a las preguntas en un tiempo específico.

5) Desarrollo de la partida: El servidor de la aplicación selecciona varias preguntas desde una base de datos, en donde las preguntas fueron previamente cargadas, tanto las preguntas como sus opciones de respuesta se escogen de forma aleatoria para luego ser enviadas a la aplicación. Las preguntas se presentan una a la vez al televidente, quien tiene un tiempo máximo en segundos para elegir una respuesta. Según el tiempo transcurrido y la validez de la respuesta seleccionada, la aplicación calcula los puntos que le serán otorgados al televidente. Si el tiempo de respuesta llega a cero o la misma es incorrecta el televidente no sumará puntos.

6) Presentación de resultados: Al finalizar la partida, se presenta al televidente un resumen indicando el puntaje total acumulado, las respuestas elegidas y el tiempo transcurrido en cada una. El televidente tiene la posibilidad de calificar el cuestionario para conocer su aceptación por parte de los televidentes.

7) Rankings: Presenta una lista de los diez mejores puntajes en cada programa de TV obtenidos por los otros televidentes registrados. Además, se presenta una lista de los programas de TV con los puntajes más altos obtenidos.

B. Arquitectura

La arquitectura empleada para el desarrollo del juego se compone de tres capas básicas (Fig. 4): La primera, una capa de datos donde será almacenada toda la información a ser consumida por el servidor, la segunda, una capa de servicio que contiene todos los servicios REST y las reglas de negocio, la tercera es la capa de aplicación donde se encuentran todas las peticiones al servidor y la interfaz gráfica desarrollada para el middleware *Ginga*.

Fig. 4. Arquitectura del juego *QuizTV*. Fuente: Elaboración propia.

A continuación, se detallan los principales componentes que forman parte de cada una de las capas de la arquitectura:

1) Base de datos: Consiste en una base de datos relacional, en la cual se almacenan todos los datos que utiliza la aplicación.

2) Servidor: Implementa mediante el marco de desarrollo de código abierto *Spring* para el lenguaje Java, aquí se establece la comunicación con la base de datos, almacenamiento y recuperación de datos de acuerdo a las peticiones de la aplicación mediante servicios REST.

3) Aplicación: Dentro de la capa de aplicación se encuentra el consumidor de servicios que esta implementado con los lenguajes NCL y Lua, en este componente se consumen los servicios REST publicados por el servidor. Por otro lado, se encuentra el componente gráfico encargado de toda la parte

visual para el televidente. Finalmente, el componente gestor de eventos controla las pulsaciones de las teclas del control remoto, inicio y finalización del juego, entre otros.

Una vez definida la arquitectura y funcionalidad de la aplicación, en la siguiente sección se detalla el desarrollo de cada uno de los componentes en el lenguaje de programación definido, así como el uso de la metodología *Ágil-TVDi*.

IV. Desarrollo de la Aplicación

Esta sección presenta la metodología empleada para el desarrollo de la aplicación, las herramientas utilizadas, así como los aspectos de programación, interfaces gráficas y pruebas finales.

A. Metodología

Para el desarrollo se adoptó la metodología *Ágil-TVDi*. Esta se basa en la metodología ágil *SCRUM* y define cada uno de los pasos a seguir para potenciar el proceso de análisis, diseño y desarrollo de aplicaciones para la TVDi [20]. Esta metodología tiene tres fases: la fase inicial que consiste en la identificación de requerimientos, la fase de producción que define las actividades de desarrollo, y la fase final para un cierre formal del proyecto.

De la fase inicial se obtuvo un documento de especificación de software, que detalla cada uno de los requerimientos funcionales y no funcionales de la aplicación, especificaciones técnicas y una lista de producto, que consistía en una serie de tareas a realizarse para llevar a cabo la aplicación. En la fase de producción, se definió el equipo de trabajo, las herramientas a ser utilizadas y el detalle de las actividades en base a la lista de producto de la fase inicial. Por último, en la fase final se realizaron diferentes pruebas de la aplicación y un informe técnico para el cierre formal del proyecto.

B. Herramientas

Dentro de la fase de producción las herramientas seleccionadas fueron: *Eclipse* como entorno de desarrollo, *NCL* y *Lua* como lenguajes de programación. *Set-Top-Box virtual* que trae preinstalado el entorno *Ginga* y se ejecuta como máquina virtual en *VMWare Workstation* y la base de datos *MySQL*. Para el control de las tareas de los programadores se utilizó la versión libre de *Trello* [21].

C. Desarrollo

El proceso de desarrollo de la aplicación *QuizTV*, fue llevado a cabo mediante varias iteraciones (*sprints*) recomendadas por la metodología. En cada iteración se planificaba a detalle cada una de las actividades y la manera en que serían ejecutadas, al terminar la iteración se revisaban los avances con el coordinador del proceso de desarrollo (*Scrum Master*) y se anotaba el objetivo alcanzado y los problemas encontrados, para ser discutidos y solventados en la siguiente iteración. A continuación, se describe brevemente como fueron diseñadas y programadas las diferentes secciones de la aplicación, dándole la funcionalidad inicialmente propuesta.

1) Entorno NCL: El lenguaje NCL fue usado para definir las diferentes regiones, descriptores y puertos de entrada para el contenido televisivo transmitido, los *scripts* del juego y los íconos que se muestran durante la transmisión. El juego está controlado en su mayoría por *scripts* Lua debido a que, al ser un lenguaje imperativo y estructurado, permite mejorar la calidad y tiempo de desarrollo reutilizando y haciendo más modular el código. Además, reduce el costo de mantenimiento y optimiza el tiempo de los programadores.

2) Control principal: Gestiona la apertura y cierre de ventanas, estructuras de datos y eventos del juego, el módulo se denomina *Main.lua*. Este módulo define una estructura que almacena la información personal del televidente que utiliza la aplicación (nombres, ciudad, país, avatar, logros, etc.) así como la información de las partidas jugadas (categoría, programa, puntaje, tiempo, respuestas elegidas, etc.). Estas estructuras pueden ser accedidas desde diferentes módulos del juego. Además, el módulo controla la apertura y cierre de las diferentes ventanas recibiendo como parámetro los eventos que surgen de la interacción entre el televidente y el juego (Fig. 5).

```
function handler(evt)
  if is_login_open then
    window_login.visualizar(evt)
  elseif is_registro_open then
    window_registro.visualizar(evt)
  elseif is_principal_open then
    window_principal.visualizar(evt)
  elseif is_selectcuestionario_open then
    window_selectcuestionario.visualizar(evt)
  elseif is_juego_open then
    window_juego.visualizar(evt)
  elseif is_resultados_open then
    window_resultados.visualizar(evt)
  elseif is_ranking_open then
    window_ranking.visualizar(evt)
  elseif is_perfilusuario_open then
    window_perfilusuario.visualizar(evt)
  end
end
```

Fig. 5. Estructura de la función *handler* en el módulo *Main.lua*. Fuente: Elaboración propia.

3) Canal de retorno: La aplicación usa un canal de retorno que permite su comunicación con el servidor en ambos sentidos. Los datos que se envían desde la aplicación al servidor son por ejemplo: login de usuarios, registro de nuevos usuarios e información de una partida jugada por el televidente. Por otro lado, el servidor envía a la aplicación información de acuerdo a la solicitud realizada mediante un módulo en Lua denominado *RESTController.lua*, que se conecta con la capa de servicios *REST* del servidor (Fig. 6). Para gestionar los eventos de conexiones *TCP*, se desarrolló un módulo adicional llamado *ConexionTCP.lua*, que especifica la dirección IP del servidor y el puerto de conexión, define una función para enviar la solicitud, recibir la respuesta en formato *JSON* y otra función encargada de verificar la disponibilidad del servidor antes de enviar una nueva petición.

```lua
function registrarUsuario(nombre,   apellido,
                          username, email,
                          password, pais,
                          ciudad,   fechaNacimiento,
                          avatar)
    local peticion ='/registrar?'..
                    'nombres='..nombre..
                    '&apellidos='..apellido..
                    '&username='..username..
                    '&email='..email..
                    '&password='..password..
                    '&pais='..pais..
                    '&ciudad='..ciudad..
                    '&fechaNac='..fechaNacimiento..
                    '&avatar='..avatar
     local response = ConexionTCP.enviarPeticion(peticion)
     return response.estado
end
```

Fig. 6. Función que envía los datos de registro de un nuevo usuario desde el módulo *RESTController.lua* al servidor. Fuente: Elaboración propia.

4) Gestor de componentes gráficos: Los componentes gráficos usados en la aplicación fueron gestionados por un módulo implementado en *Lua*, denominado *ToolsGUI.lua*. Este módulo define funciones que permitieron dibujar los diferentes elementos de la pantalla. También define funciones que refrescan la pantalla y controlan la actualización de los elementos gráficos (Fig. 7). Adicionalmente, define funciones que permiten modificar de forma automática el tamaño del texto y el número de líneas en que se debe dividir cuando el espacio es reducido. Además, este módulo emplea una librería que gestiona la parte gráfica a más bajo nivel denominada *canvas.c* [22].

```lua
JUEGO = {
  BKG = {
    FONDO = {"./media/WJuego/background.png"}
  },
  BTN = {
    ROJO     = {"./media/WJuego/btnRojoFN.png",
                "./media/WJuego/btnRojoFS.png"},
    VERDE    = {"./media/WJuego/btnVerdeFN.png",
                "./media/WJuego/btnVerdeFS.png"},
    AMARILLO = {"./media/WJuego/btnAmarilloFN.png",
                "./media/WJuego/btnAmarilloFS.png"},
    AZUL     = {"./media/WJuego/btnAzulFN.png",
                "./media/WJuego/btnAzulFS.png"}
  }
}
```

Fig. 7. Estructura que almacena los paths de las imágenes que son mostradas en la pantalla de juego. Fuente: Elaboración propia.

5) Gestor de imágenes: Se creó un módulo denominado *ImagenesGUI.lua*, que almacena los *paths* de todas las imágenes que forman parte de las interfaces gráficas, por ejemplo: imágenes de fondo de pantalla, imágenes para los botones, etc. La finalidad de este módulo es separar la declaración de imágenes del código que gestiona cada interfaz gráfica lo que permitió controlar posibles errores generados, evitando la finalización inesperada de la aplicación.

D. Diseño de interfaces gráficas

Para el diseño de las interfaces gráficas, se contó con la ayuda de un diseñador gráfico con la finalidad de presentar un entorno visualmente atractivo para el televidente. El juego contó con ocho interfaces gráficas, cada una se gestionó en archivos *.lua* diferentes, creando una estructura donde se definen cada uno de sus componentes gráficos. Esta estructura es enviada al módulo gestor de componentes gráficos, que se encarga de mostrar cada elemento en la pantalla según las propiedades definidas.

Las imágenes usadas en el juego fueron exportadas en formato PNG (*Portable Network Graphics*), debido a la transparencia del formato. La mayoria de las imágenes fueron elaboradas por el diseñador del proyecto, considerando los colores que pertenecen al espectro cromático de los fotones digitales de las pantallas de TV y los botones de movimiento del control remoto (Fig. 8).

Fig. 8. Logo de la aplicación en formato PNG. Fuente: Elaboración propia.

Los botones del control remoto cumplen funciones diferentes en cada interfaz, por lo que las interfaces controlan los eventos de tipo *key* de forma independiente. La selección de opciones por parte del televidente se centró en el uso de los cuatro botones que representan los colores del control remoto: rojo, verde, amarillo y azul (Fig. 9). De esta manera, el televidente puede seleccionar la opción a elegir presionando directamente el botón con el color correspondiente, evitando tener que usar las flechas para ubicarse en la opción deseada y pulsar *OK* para confirmar la selección (esta posibilidad también está disponible en el juego).

Fig. 9. a) Grupo de botones mostrados en la pantalla principal. b) Grupo de botones que representan a las opciones de respuesta de una pregunta durante una partida. c) Grupo de botones mostrados al finalizar una partida. Fuente: Elaboración propia.

Tambien se implementaron algunas ayudas para el televidente respecto a las funcionalidades que están disponibles y la forma en la que accede, debido a que estas ayudas juegan un papel importante en la usabilidad del juego. Por este motivo,

en algunas pantallas se muestra información de los botones del control remoto que pueden ser usados y una breve descripción de la acción que ejecutan. En la pantalla donde el televidente responde las preguntas, se implementó un temporizador que refleja el tiempo restante que tiene para elegir una opción, para lo cual se ejecutaron, de forma secuencial, dos ficheros Lua distintos. En primera instancia, se ejecuta el código que controla la lógica de la pantalla del juego y a continuación, se ejecuta el código del temporizador el cual controla el tiempo mediante la función *event.timer()* y grafíca en forma textual, el tiempo restante sobre la ventana controlada por el código de la lógica del juego (Fig. 10).

```
function paintTexto(text, fontFamily, size, fontColor,
                    pos_x, pos_y)
  canvas:attrColor(fontColor)
  canvas:attrFont(fontFamily,size,"bold")
  canvas:drawText(pos_x,pos_y,text)
end

function paintImagen(pos_x, pos_y, src_img)
  canvas:compose(pos_x,pos_y,canvas:new(src_img))
end
```

Fig. 10. Funciones que permiten la gestión de componentes gráficos para textos e imágenes definidas en el módulo *ToolsGUI.lua*. Fuente: Elaboración propia.

E. Pruebas con usuarios finales

Luego de finalizar la etapa de desarrollo se llevaron a cabo diferentes pruebas del juego con usuarios reales, cuya finalidad era detectar posibles errores y mejorar aspectos tanto de la interfaz gráfica como la lógica del juego. Los usuarios de prueba fueron personas que colaboran en distintos proyectos del departamento de Ciencias de la Computación de la Universidad de Cuenca. La aplicación fue desplegada en un Set-Top-Box virtual emulado por una máquina virtual, la base de datos y el servidor fueron ejecutados localmente sobre una misma red. Finalmente, se procedió a elaborar el informe final de actividades realizadas según la recomendación de la metodología *Ágil-TVDi*.

Por otro lado, el juego fue presentado en una exposición abierta a estudiantes universitarios y de colegios. Los usuarios completaron satisfactoriamente las trivias que eligieron a su gusto, se mostraron entretenidos y no tuvieron ningún problema en la navegación por la aplicación. Las interfaces gráficas del juego fueron lo que mas llamo su atención.

V. RESULTADOS

Luego del desarrollo de la aplicación, a continuación se indican los resultados obtenidos con la presentación de las principales interfaces que se encuentran en la aplicación siguiendo el flujo de una partida del juego, desde la pantalla principal, pasando por la selección de parámetros de juego, el desarrollo de una partida y la presentación de resultados.

A. Menú principal

El menú principal (Fig. 11) se muestra una vez que el televidente haya iniciado sesión con sus credenciales o se haya registrado en el caso de ser usuarios nuevos. En la pantalla se presenta la información básica del televidente, su avatar y, además, presenta una lista de los cuestionarios con los puntajes más altos que ha obtenido en partidas anteriores. El televidente tiene la posibilidad de elegir entre cuatro opciones de un menú, esta selección se puede realizar oprimiendo directamente las teclas de colores o flechas del control remoto.

Fig. 11. Pantalla principal del juego. Fuente: Elaboración propia.

B. Selección de parámetros del juego

Para iniciar una nueva partida es necesario seleccionar los siguientes parámetros (Fig. 12): Una categoría de programas de TV, automaticamente la aplicación lista todos los programas que están disponibles para la categoría seleccionada, selección del programa, el modo de juego y presionar el botón comenzar o regresar a las opciones anteriores.

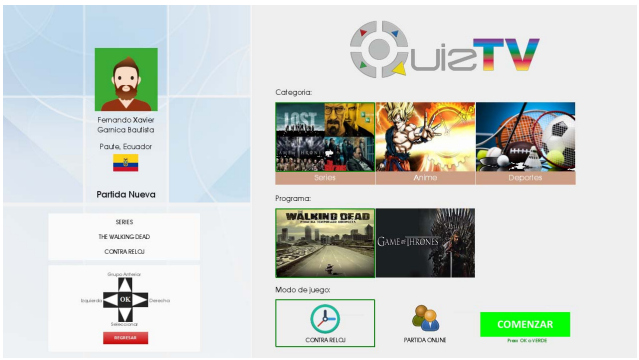

Fig. 12. Pantalla para selección de parámetros de juego. Fuente: Elaboración propia.

C. Desarrollo de la partida

Una vez iniciada una partida, se presenta al televidente un total de 10 preguntas. En cada pregunta se tiene un cronómetro que indica el tiempo disponible para elegir una opción y una imagen relacionada a la pregunta. Las opciones de respuesta se muestran en cuatro rectángulos, cada uno presenta un color que hace referencia a uno de los botones de colores del control remoto para su selección directa. Si el jugador acierta, se mostrará en la parte izquierda de la pantalla el puntaje obtenido en función del tiempo que le tomó contestar (Fig. 13), caso contrario, en la parte derecha de la ventana se mostrará un

puntaje de cero cuando la respuesta elegida sea incorrecta o el tiempo disponible para seleccionar una opción se haya agotado. Posteriormente, se presentará la siguiente pregunta.

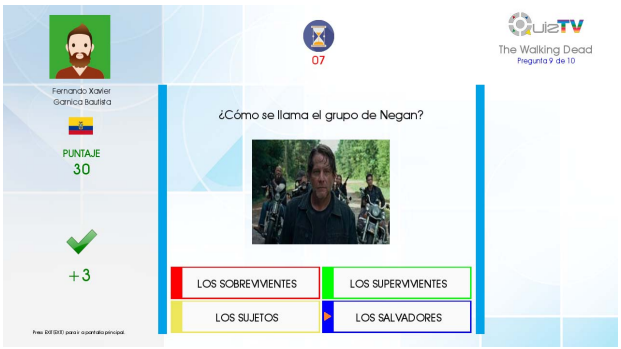

Fig. 13. Pantalla de desarrollo de la partida. Fuente: Elaboración propia.

D. *Finalización de la partida*

Al terminar de responder todas las preguntas del cuestionario, se presenta una pantalla (Fig. 14) que muestra la información de la partida y el puntaje obtenido. Además, se permite al televidente realizar una revisión de las respuestas que eligió para cada pregunta durante la partida, la revisión no muestra las respuestas correctas de las preguntas, únicamente presenta las que el televidente seleccionó, esto con la finalidad de generar curiosidad en el jugador y motivarlo a volver a ver el programa y así en una siguiente partida acertar a las preguntas fallidas. Además, el televidente tiene la posibilidad de calificar el cuestionario mediante un número de estrellas de entre cero y cinco.

Fig. 14. Pantalla para presentación de resultados de la partida jugada. Fuente: Elaboración propia.

El televidente tiene la posibilidad de jugar todas las partidas que guste sobre el programa de su preferencia. En esta primera versión de la aplicación, el modo de juego disponible es ”contra reloj” y las categorías así como los programas, ya se encuentran registrados en la base de datos del servidor local. La interfaz gráfica, así como todo el desarrollo de la funcionalidad del juego trajo varios aciertos que han sido descritos en esta sección, sin embargo, se presentaron algunos inconvenientes que deben ser considerados para futuras versiones o nuevas implementaciones para TVDi, los mismos que serán descritos en la siguiente sección.

VI. Inconvenientes Encontrados

Entre los aspectos que causaron ciertos inconvenientes al momento del desarrollo de la aplicación estuvo el diseño de algunas interfaces gráficas debido a que, en las versiones iniciales del juego, luego de las pruebas con los usuarios, se detecto que había sobrecarga de componentes, provocando cansancio visual en los televidentes. De esta manera se tuvieron que hacer rediseños, lo que modificó la planificación inicial.

Otro inconveniente fue el tiempo que les tomó a los desarrolladores codificar todos los diseños de las pantallas, debido a que era necesario especificar la posición de cada componente dentro de las pantallas pixel por pixel, debido a que para la programación con NCL y Lua no se cuenta con un marco de trabajo que optimice el control gráfico y no se han creado herramientas de diseño de interfaces gráficas que permitan únicamente arrastrar elementos disponibles en una paleta a la ventana e ir observando los resultados al mismo tiempo. Además, dentro del mismo contexto gráfico, se detectó que la saturación con el color negro no es manejado de manera adecuada por la librería *NCLua*, lo que provoca que la imagen se vea incompleta en las zonas saturadas. El problema fue solucionado aplicando un proceso de corrección de saturación a las imágenes usando herramientas de software para diseño gráfico externas.

Gestionar los componentes gráficos desde *Lua* para el redimensionamiento de imágenes en diferentes tamaños de pantallas de TV, también causó un gran inconveniente, ya que la librería gráfica *canvas.c* no provee mecanismos que permitan dar solución al problema, los únicos componentes que se pueden redimensionar son aquellos definidos por la librería (texto, líneas, etc.). El problema se agudiza cuando se desea redimensionar imágenes en cualquier formato (PNG, JPG, etc.), por lo que la aplicación fue desarrollada para pantallas de 1280x720, como solución inmediata. Esto no se presenta cuando la gestión de componentes gráficos se realiza enteramente desde NCL, sin embargo esta técnica dificulta el control gráfico y las líneas de código aumentan de forma descontrolada, provocando que el mantenimiento de aplicaciones extensas sea una tarea muy complicada.

VII. Conclusiones y Trabajos Futuros

El presente trabajo presentó un proceso de análisis, diseño y desarrollo de un juego para la TVDi, usando el middleware *Ginga* en el que se consideró: la metodología de desarrollo Agil-TVDi, la definición de las herramientas necesarias para el desarrollo y planificacion de cada una de las actividades, todo el proceso de desarrollo sin dejar de lado el diseño de interfaces gráficas y el envio de recepción de información de acuerdo a las peticiones del usuario, lo que es fundamental en una aplicacion de tipo interactiva. Así tambien, se expuso la experiencia adquirida en todo el proceso, las facilidades de los entornos de programación para TV como NCL y Lua,

para finalmente mostrar los diferentes desafios encontrados y la manera que fueron solventados.

La lógica del juego fue implementada en su mayoría en Lua, debido a que permite obtener mayor modularidad en el código y mejora los tiempos del equipo de desarrollo. Entre los principales inconvenientes estaba el redimensionamiento automático de las imágenes, para diferentes resoluciones de pantalla, fue necesario disminuir el alcance del juego para ir dando solución a las dificultades presentadas con los componentes gráficos. Sin embargo se logró culminar una aplicación bastante intuitiva y amigable con los televidentes, todo el esfuerzo empleado en la parte gráfica fue de gran utilidad para conseguir un producto atractivo, además, la navegación entre las opciones de respuesta en la pantalla del juego, permiten que el televidente pueda moverse entre dichas opciones usando las flechas de movimiento del control remoto o presionando directamente el color correspondiente a la opción seleccionada.

Como conclusión final, y en base a la experiencia obtenida, se considera necesaria la creación de marcos de trabajo estandarizados que permitan gestionar el diseño de interfaces gráficas a un nivel más alto. La finalidad del marco de trabajo es agilizar los procesos de diseño haciendo que los desarrolladores únicamente tengan que arrastrar desde una paleta de elementos gráficos los diferentes componentes, evitando tener que crear dichos componentes desde el código fuente, lo que permitiría optimizar recurso y utilizar de mejor manera el tiempo de diseño en implementar mejores y nuevas funcionalidades a los aplicativos.

Como trabajos futuros se plantea la implementación de dispositivos de segunda pantalla que sustituyan el control remoto para tareas básicas como el ingreso de texto, selección de opciones entre otras, que a pesar de ser bastante sencillas se tornan complejas para el televidente. De esta manera se puede dar mayor complejidad a las aplicaciones y mejorar la experiencia e interactividad de los televidentes. Además, se plantea la incorporación del modo de juego online, que consiste en llevar a cabo una partida con dos televidentes conectados sobre un mismo cuestionario, el ganador será aquel que haya obtenido el mayor puntaje. Finalmente, se propone realizar pruebas de la aplicación en entornos reales y con diferentes tipos de televidentes.

AGRADECIMIENTOS

Este trabajo es parte del proyecto "Análisis de Plataformas para el desarrollo de aplicaciones para la Televisión Digital Terrestre". Se agradece a la Dirección de Investigación de la Universidad de Cuenca (DIUC) por el apoyo y financiamiento brindado.

REFERENCES

[1] M. S. Alencar, W. T. Lopes, and F. Madeiro, "History of television in brazil," in *Telecommunications Conference (HISTELCON), 2010 Second IEEE Region 8 Conference on the History of*. IEEE, 2010, pp. 1–4.

[2] K. Alic, M. Zajc, M. Tkalcic, U. Burnik, and J. Tasic, "Development of interactive television t-learning course," in *Electrotechnical Conference, 2008. MELECON 2008. The 14th IEEE Mediterranean*. IEEE, 2008, pp. 139–144.

[3] S. Morris and A. Smith-Chaigneau, *Interactive TV standards: a guide to MHP, OCAP, and JavaTV*. CRC Press, 2012.

[4] D. Véras, I. Ibert, H. Barros, M. Silva, and E. Costa, "A solution for personalized t-learning applications integrated with a web educational platform," in *Proceedings of the 2011 ACM Symposium on Applied Computing*. ACM, 2011, pp. 1187–1193.

[5] L. Xiang and Q. Wang, "Influence of interactive television on electronic commerce," in *Management of e-Commerce and e-Government, 2009. ICMECG'09. International Conference on*. IEEE, 2009, pp. 513–516.

[6] J. Santos, E. Ratamero, J. P. Arruda, M. Dantas, M. L. Sanchez, and D. C. M. Saade, "Rummitv: An interactive game for the brazilian digital tv system," in *Proc. 26th Symp. Brasileiro Telecomun.*, 2008, pp. 1–6.

[7] I. A. Quintero, M. M. Rodriguez, C. O. Barajas, J. V. P. Ceron, P. M. Rodriguez, and A. N. Cadavid, "Kroster-mhp game for digital tv. developing process, design, and programming considerations against technical issues," *IEEE Revista Iberoamericana de Tecnologias del Aprendizaje*, vol. 8, no. 4, pp. 166–175, 2013.

[8] M. Takada and M. Saito, "Transmission system for isdb-t," *Proceedings of the IEEE*, vol. 94, no. 1, pp. 251–256, 2006.

[9] M. C. Farias, M. M. Carvalho, and M. S. Alencar, "Digital television broadcasting in brazil," *IEEE multimedia*, vol. 15, no. 2, pp. 64–70, 2008.

[10] "Official site of ginga middleware," http://www.ginga.org.br/en, accessed: 2018-04-19.

[11] G. L. d. Souza Filho, L. E. C. Leite, and C. E. C. F. Batista, "Ginga-j: The procedural middleware for the brazilian digital tv system," *Journal of the Brazilian Computer Society*, vol. 12, no. 4, pp. 47–56, 2007.

[12] L. F. G. Soares, R. F. Rodrigues, and M. F. Moreno, "Ginga-ncl: the declarative environment of the brazilian digital tv system," *Journal of the Brazilian Computer Society*, vol. 12, no. 4, pp. 37–46, 2007.

[13] S. d. T. del Ecuador, "Informe para la definición e implementación de la televisión digital terrestre en ecuador," 2010.

[14] J. Crespo, A. Tello, V. Saquicela, K. Palacio-Baus, and M. Espinoza, "Tv program recommender using user authentication on middleware ginga," in *Ecuador Technical Chapters Meeting (ETCM), 2017 IEEE*. IEEE, 2017, pp. 1–6.

[15] A. C. Paredes, N. M. Tonguino, G. F. Olmedo, and F. R. Acosta, "Performance analysis on return channel for interactive digital tv isdb-tb system," in *Communications (LATINCOM), 2012 IEEE Latin-America Conference on*. IEEE, 2012, pp. 1–4.

[16] D. Villamarín, M. A. Illescas, G. Olmedo, and R. L. Cueva, "Generating a transport stream for digital terrestrial television system in conformance with isdb-tb standard," in *Communications and Computing (COLCOM), 2013 IEEE Colombian Conference on*. IEEE, 2013, pp. 1–5.

[17] I. Bernal and D. Mejía, "Building a basic hardware and software infrastructure for developing ginga-ncl interactive applications," in *Applications and Usability of Interactive TV*. Springer, 2015, pp. 74–89.

[18] A. N. d. S. Junior, A. C. S. Souza, and L. C. M. d. Santos, "Development of games for set-top-boxes with brazilian's middleware ginga-ncl," in *Proceedings of the 2009 VIII Brazilian Symposium on Games and Digital Entertainment*. IEEE Computer Society, 2009, pp. 214–219.

[19] R. M. Segundo, J. C. F. da Silva, and T. A. Tavares, "Athus: A generic framework for game development on ginga middleware," in *Games and Digital Entertainment (SBGAMES), 2010 Brazilian Symposium on*. IEEE, 2010, pp. 89–96.

[20] V. Mora, E. Sari, E. Cabrera, M. Mejía, and L. Muñoz, *Manual Técnico - Metodología de Desarrollo Ágil-TVDi*, Universidad de Cuenca, 03 2018.

[21] "Trello," https://trello.com/, accessed: 2018-04-19.

[22] "Librería canvas para nclua," https://github.com/TeleMidia/nclua/blob/master/nclua/canvas.c, accessed: 2018-04-19.