UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

TRABAJO FIN DE GRADO

Desarrollo de módulos software para la interacción entre herramientas de medición de audiencias y plataformas de gestión de contenidos basadas en Digital Signage

Titulación: Grado en Ingeniería en Tecnologías de la Telecomunicación
Autor: Samuel Ángel Martel Santana
Tutores: Dionisio Rodríguez Esparragón
Luis Domínguez Quintana

Fecha: Diciembre 2014

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

TRABAJO FIN DE GRADO

Desarrollo de módulos software para la interacción entre herramientas de medición de audiencias y plataformas de gestión de contenidos basadas en Digital Signage

HOJA DE FIRMAS

Alumno/a

Fdo.: Samuel Angel Martel Santana

Tutor/a

Fdo.: Dionisio Rodríguez Esparragón

Tutor/a

Fdo.: Luis Domínguez Quintana

Fecha: Diciembre 2014

UNIVERSIDAD DE LAS PALMAS
DE GRAN CANARIA

ESCUELA DE INGENIERÍA DE TELECOMUNICACIÓN Y ELECTRÓNICA

TRABAJO FIN DE GRADO

Desarrollo de módulos software para la interacción entre herramientas de medición de audiencias y plataformas de gestión de contenidos basadas en Digital Signage

HOJA DE EVALUACIÓN

Calificación: _____

Presidente

Fdo.:

Vocal

Fdo.:

Secretario/a

Fdo.:

Fecha: Diciembre 2014

Índice de contenidos

Índice de figuras

Índice de tablas

Capítulo 1. Introducción

En este capítulo se detalla el propósito de este Trabajo Fin de Grado, y se describen los objetivos que se desean conseguir con el mismo.

1.1 Antecedentes

La tecnología Digital Signage avanza con gran rapidez y cada vez es más frecuente encontrar pantallas en lugares públicos que se emplean con fines informativos o publicitarios.[1] Uno de los mayores retos de esta tecnología es medir la eficiencia de la comunicación realizada con estos dispositivos así como lograr la interacción con el público. Fabricantes de software y hardware desarrollan herramientas cada vez más precisas para la detección de los individuos que miran a estas pantallas, herramientas que permiten obtener datos sobre el tiempo de exposición, género de la audiencia y su rango de edad, entre otros.[2][3]

La integración de estas herramientas de medición con las plataformas que gestionan el contenido que se emite resulta clave para el desarrollo de nuevas formas de interacción con el público objetivo y una gestión adaptada al perfil de la audiencia. Para ello los fabricantes de las herramientas de medición ofrecen paquetes de desarrollo (Software Development Kits) que proporcionan recursos para lograr esta comunicación en tiempo real entre ambas soluciones, la herramienta de medición y el software de gestión de contenidos.

Con este Trabajo Fin de Grado se pretende afrontar el diseño de módulos software que actúen como interfaces de comunicación entre el software de gestión de contenidos Scala, ya disponible en el Laboratorio de Medios de Producción de TV de la EITE, y la herramienta de medición.

Scala es pionero de la industria de cartelería digital y a la fecha tiene más de 300,000 licencias en uso por todo el mundo.[4] Entre los miles de clientes de la cartelería digital de la empresa, se incluyen a Rabobank, IKEA, Burger King, T-Mobile, Virgin MegaStore, EuroDisney, McDonalds, Warner Brothers, Repsol, Shell, NorgesGruppen, etc. Scala tiene sus oficinas generales en Philadelphia, EEUU, tiene oficinas en los Países Bajos, Noruega, Alemania, Dinamarca, Suecia, Reino Unido, Japón y tiene más de 450 asociados en más de 60 países. En febrero de 1999, Scala fue nombrada por la revista *Animación* para estar entre las 13 mejores empresas de fabricantes de software 2D.[5]

Para ello se llevarán a cabo las siguientes tareas preliminares:

- Analizar, en concreto, las capacidades de la plataforma Scala para comunicarse con soluciones software externas y definir protocolos para la gestión eficiente de la comunicación en base a condiciones externas.
- Analizar el estado del arte sobre las soluciones de detección y medición de audiencias, y su uso en entornos de Comunicación y Publicidad Dinámica basados en Digital Signage.
- Elegir la solución de medición que ofrezca mejores prestaciones y analizar, en concreto, sus posibilidades de integración en la plataforma de gestión de contenidos Scala.

Tras llevar a cabo lo anterior pasaremos a diseñar un software que sirva de nexo de unión entre ambas herramientas.

1.2 Objetivos

El objetivo central será ejecutar la integración software de la herramienta de medición elegida con la plataforma de gestión Scala. Una vez lograda la interacción entre ambas soluciones se llevará a cabo su test en entornos de producción real con el fin de lograr una gestión eficiente e inteligente de contenidos a partir de los datos de medición obtenidos.

Capítulo 2. Digital Signage

En este capítulo se estudiará el concepto de Digital Signage; contenidos, tecnologías involucradas, mercados, aplicaciones,…

2.1 Introducción

La Señalización Digital (*Digital Signage* en inglés), normalmente conocida como señalización digital dinámica o señalización digital multimedia, es el uso de contenidos digitales emitidos a través de pantallas como monitores LCD, pantallas de plasma o un panel de LED. La señalización digital suele utilizarse en letreros de orientación, exposiciones, arte público, marketing y publicidad exterior.

La señalización digital, en adelante DS, usa tecnologías como LCD, LED y proyecciones para reproducir contenidos como imágenes, video, video por streaming e información, y puede ser encontrado en lugares públicos, transportes públicos, museos, estadios, tiendas, hoteles, restaurantes, edificios corporativos, etc.

Figura 2.1. Señalización Digital en el aeropuerto mallorquín de Son Sant Joan, Mallorca

La DS usa sistemas de gestión de contenidos y sistemas de distribución de medios digitales que se pueden ejecutar desde cualquier ordenador personal y servidor o proveedor de alojamiento de medios de comunicación regional o nacional.

Dado que el contenido de la DS puede ser actualizado con frecuencia y facilidad, y también debido a las capacidades interactivas disponibles a través del empleo de dispositivos, tales como pantallas táctiles integradas, detección de movimiento y dispositivos de captura de imagen que permiten a estas formas de señalización saber quién y cómo los usuarios interactúan con ellos, la DS está ganando aceptación como una alternativa a la señalización estática.

Uno de los usos específicos de la DS es la publicidad exterior en la que se muestran contenidos de vídeo, anuncios y mensajes en las pantallas con el objetivo de mostrar una información determinada, en lugares y a consumidores específicos, en momentos específicos.

2.2 Contenido

El contenido puede ser cualquier cosa designada para mostrarse en las pantallas. Desde texto a imágenes, animaciones, video, audio e interfaces. Instalar la DS es caro respecto a la señalización estática y necesita contenido que se considere útil para el usuario y así recuperar la inversión realizada.[6]

Hay muchos ejemplos de la utilización con éxito de la DS y diseño de contenidos digitales, como la terminal internacional Tom Bradley recientemente completada en el aeropuerto de Los Ángeles[7], la pasarela en The Dubai Mall[8] (Figura 2.2) y muchas otras instalaciones no publicitarias que mejoran el ambiente y que ofrecen información valiosa para los usuarios.

Figura 2.2. Pasarela en The Dubai Mall

El diseño de contenidos se realiza normalmente a través de una empresa especializada, tales como Digital Kitchen, Arsenal Media, Unified Field, Gallagher and Associates, OpenEye, etc.

2.3 Gestión de Contenidos

Hay un gran número de diferentes proveedores de software disponibles, incluyendo SCALA, Daktronics, S|N|A, Planar, Nanolumens y muchos más. En muchas aplicaciones de DS el contenido debe ser actualizado regularmente para asegurarse de que se muestran los mensajes correctos. Esto se puede hacer de forma manual cómo y cuando sea necesario, a través de un sistema de programación, mediante el uso de una fuente de datos de un proveedor de contenidos, o una base de datos de la empresa.

2.4 TECNOLOGIA

La DS se basa en una variedad de hardware para mostrar el contenido. Los componentes de una instalación típica incluyen una o más pantallas, uno o más reproductores multimedia y un servidor de gestión de contenidos. A veces dos o más de estos componentes están presentes en un único dispositivo, pero normalmente hay una pantalla de visualización, un reproductor de medios, y un servidor de gestión de contenido que está conectado al reproductor multimedia en una red. Un servidor de gestión de contenidos puede admitir varios reproductores multimedia y un reproductor de medios puede soportar múltiples pantallas. Los reproductores multimedia de DS se ejecutan en una gran variedad de sistemas operativos, incluyendo Windows, Linux, Android e iOS.

2.4.1 Construcción de visualización modular

Las pantallas de matriz de LED a menudo se usan como componentes de las pantallas modulares, para permitir diferentes tamaños y formas de las pantallas, y hacer más fácil el montaje y la construcción. Una pantalla modular consta de dos partes:

- Módulos de matriz de pantalla (8x8 píxeles, 16x16 píxeles, 8x16 píxeles, etc.)
- Controlador de matriz de visualización.

Por ejemplo, una pantalla de tamaño variable puede utilizar módulos de 16 LEDs de ancho y 16 LEDs de alto. Para construir una pantalla de 64 píxeles de ancho y 32 píxeles de alto, la pantalla se forma usando una construcción de cuatro módulos de ancho y dos módulos de altura.

Los módulos de matriz se pueden unir al controlador utilizando conectores de datos individuales, lo que limita el área de visualización según el número total de

conectores de datos disponibles en el controlador. Los módulos pueden comunicarse con el controlador usando un bus de datos común, y la posición del módulo de matriz para visualizar su parte de la imagen global se le asigna mediante un número de identificación a través del bus de datos.

La reutilización de códigos de posición permite mostrar la misma información en más de un módulo de la matriz. De esta manera, una pantalla de doble cara o cuatro caras se puede construir utilizando un controlador de pantalla de matriz única, y la reutilización de todos los códigos de posición de los módulos en cada cara de la pantalla.

Las pantallas LCD o de plasma estándar también se pueden combinar de esta manera mediante el uso de un controlador especial VGA, pero por lo general hay un área de visualización inutilizable alrededor del perímetro de un panel LCD o plasma estándar que no se puede ocultar, los paneles LCD combinados tienden a tener la aparición de una imagen rota en azulejos (Figura 2.3).

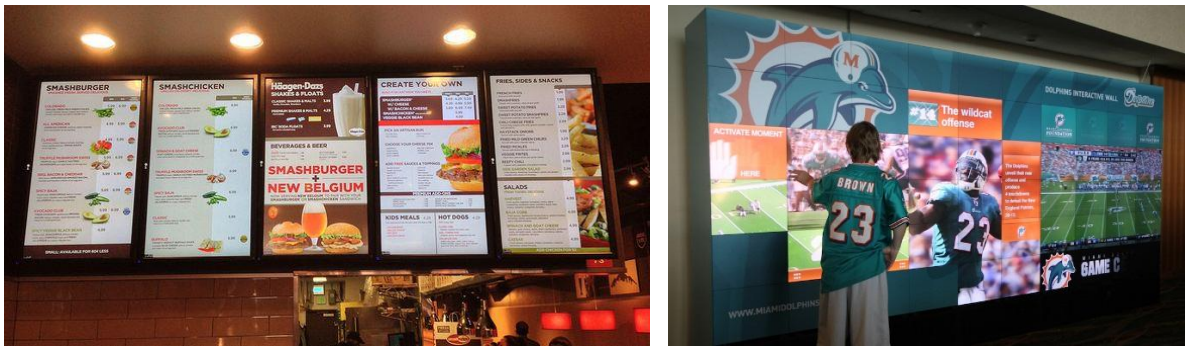

Figura 2.3. A la izquierda pantalla modular con pantallas LCD y a la derecha pantalla modular

2.4.2 Pantallas 2D y 3D

Las pantallas para la DS pueden ser pantallas LCD o de plasma, tableros LED, pantallas de proyección u otros tipos de pantalla como superficies interactivas o pantallas LED orgánicos (OLED). Se están desarrollando nuevas tecnologías para la DS en la actualidad, como pantallas tridimensionales (3D), con o sin gafas 3D (Figura 2.4), pantallas de agua, pantallas de niebla (Figura 2.5) y displays holográficos (Figura 2.6).

Figura 2.4. A la izquierda antalla TV 3D con gafas y a la derecha pantallaTV 3D sin gafas

Figura 2.5. Pantallas de agua y pantallas de Niebla

Figura 2.6. Displays holográficos

Debido a cuestiones de costes, hasta el momento muchas de estas nuevas tecnologías sólo se ha empleado para las instalaciones más pequeñas, en lugar de para grandes pantallas o redes.

La rápida caída de los precios para las grandes pantallas de plasma y LCD han dado lugar a un aumento en el número de instalaciones de DS.[9]

2.4.3 La reproducción y gestión de contenidos

El contenido audiovisual se reproduce en televisores y pantallas de una red de DS a partir de al menos un reproductor de medios (por lo general una unidad de equipo pequeño, pero también se suelen utilizar los reproductores de DVD y otros tipos de medios de reproducción).

Existen varias opciones de hardware y software, proporcionando diferentes maneras de programar y reproducir contenido. Estos van desde simples reproductores portátiles que no estén conectados a la red, que puede dar salida a presentaciones de diapositivas JPG o bucles de vídeo MPEG-2 a las redes complejas que consisten en múltiples reproductores y servidores que ofrecen el control de todas pantallas instaladas en muchos lugares de una única ubicación. Los primeros son ideales para pequeños grupos de pantallas que se pueden actualizar a través de una unidad flash USB, una tarjeta SD o un CD. Otra opción es el uso de una red digital de publicidad (Digital Advertising Network, D.A.N.), reproductores que se conectan directamente al monitor y a internet, a una WAN (Wide Area Network), o a una LAN (Local Area Network). Esto permite al usuario final la posibilidad de gestionar múltiples reproductores DAN desde cualquier lugar. El usuario final puede crear nueva publicidad o editar anuncios existentes y luego cargar los cambios en la DAN a través de Internet u otras opciones de red.

Si se utiliza un software más avanzado para la DS podemos crear el contenido por los reproductores multimedia y servidores de forma automática minuto a minuto, combinando datos en tiempo real, de las noticias, del tiempo, precios, horarios de transporte, etc., con contenido audiovisual para reproducir el contenido más actualizad

2.4.4 Infraestructura de la Red

Cada vez que la pantalla, el reproductor multimedia y el servidor de contenido se encuentran separados hay una necesidad de cableado de audio y vídeo entre la pantalla y el reproductor multimedia y entre el reproductor multimedia y el servidor de contenido. La conexión desde el reproductor multimedia a la pantalla es normalmente un VGA, DVI, HDMI o por componentes de conexión de vídeo. A veces, esta señal se distribuye a través de cables de categoría 5 mediante baluns[10] que permite una mayor distancia entre la pantalla y el reproductor. La conexión desde el reproductor multimedia hasta el servidor de contenido es por lo general una conexión Ethernet aunque algunas instalaciones utilizan redes Wi-Fi.

Para gestionar una red, se requiere generalmente un servidor de gestión. Éste puede ser situado en cualquier lugar, siempre y cuando esté conectado a la red de DS. Las redes de DS pueden o bien ser cerradas o abiertas a la web, lo que afectará a cómo se actualiza el contenido de las pantallas. Para las redes cerradas (sin acceso a Internet), las actualizaciones se deben hacer a nivel local a través de memorias USB, unidades de DVD u otros dispositivos 'in situ'. Las redes abiertas (con acceso a Internet) se pueden actualizar de forma remota y obtener un flujo de datos desde otras fuentes de Internet (como los canales RSS).

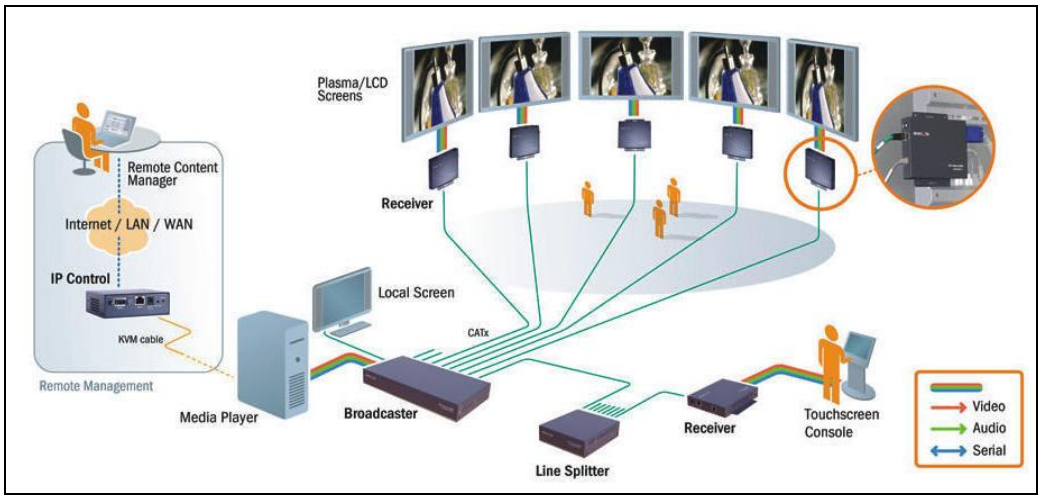

Figura 2.7. Ejemplo de infraestructura de una red

2.4.5 Otras tecnologías

La DS puede interactuar con los teléfonos móviles. Usando la mensajería SMS y el Bluetooth, algunas redes están aumentando la interactividad de la audiencia. Los sistemas de SMS se pueden utilizar para enviar mensajes en las pantallas, mientras que el Bluetooth permite a los usuarios interactuar directamente con lo que ven en la pantalla. Además de la interactividad móvil, las redes también están utilizando la tecnología que integran las redes sociales. Esta tecnología permite a los usuarios finales enviar mensajes desde Twitter y Flickr, así como mensajes de texto a las pantallas.

Algunos sistemas de gestión de colas utilizan la tecnología de pantalla dividida para combinar la gestión de colas con la DS. La información requerida por la audiencia atrae la atención y el mensaje de vídeo contiguo beneficia simultáneamente. La sinergia resultante es una parte inherente de las estrategias de gestión.

2.5 Mercados y Aplicaciones

Actualmente, China es líder mundial en el número de redes de DSes desplegadas y en el número de salidas a bolsa de NASDAQ, con la mayor firma del país, Focus Media Holding, operando con más de 120.000 pantallas.[11]

Otra fuente de información sobre las pantallas de DS e impresiones (el número de veces que un espectador lee/mira la DS), es un informe presentado por Nielsen Company, el "4th *Screen Network Audience Report".* En él la compañía Nielsen identifica que las pantallas digitales en la categoría de "cuarta pantalla" en los Estados Unidos generaron más de 237 millones de exposiciones mensuales a personas mayores de 18 años. El informe identifica Screenvision, NCM, Captivate, GSTV e IndoorDirect como las empresas líderes en las pantallas de “cuarta categoría”.

Una de las empresas líderes de DS en salas de cine es Screenvision, con más de 14.400 pantallas en los EE.UU.; otro líder en el mercado "cuarta pantalla" es GSTV (Gas Station TV), que genera más de 32 millones de exposiciones cada mes. Nielsen estima que estos más de 237 millones de exposiciones se traducen en que más de la mitad (54%) de la población adulta está expuesta a un anuncio de vídeo de una red de DS durante el periodo medido.[12]

La DS se utilizan para muchos propósitos diferentes y no existe una lista definitiva. A continuación se muestran algunas de las aplicaciones más comunes de la DS:

- **Información pública** – noticias, el tiempo, tráfico e información local, como salidas de emergencia y de información al viajero.
- **Información interna –** mensajes corporativos, como artículos de salud y seguridad, noticias,…
- **Menú –** precios, fotos, ingredientes, y otra información sobre la comida que se ofrece, incluida información nutricional.
- **Publicidad –** ya sea relacionada con la ubicación de la red o publicidad genérica.
- **Mejorar la experiencia del cliente –** las aplicaciones incluyen la reducción de la percepción del tiempo de espera en las salas de espera de los restaurants y otras tiendas al por menor, las colas de los bancos, etc.

- **Mejorar el ambiente –** con pantallas interactivas, por ejemplo en el suelo, con información que se encuentran en algunos lugares de interés turístico, museos, etc.

Capítulo 3. Análisis de audiencia

En este capítulo se verán diferentes tipos de análisis de audiencias llevados a cabo según el medio de estudio.

3.1 Audiencia

El Diccionario de la Real Academia Española de la Lengua Española define audiencia como número de personas que reciben un mensaje a través de cualquier medio de comunicación. En un principio la audiencia era el conjunto de espectadores de teatro o de cualquier clase de representación pública, con la invención de la imprenta la audiencia era el público lector y, a posteriori, con la aparición de los medios de comunicación de masas como la radio y la televisión y de las industrias publicitarias, la palabra fue adquiriendo otras connotaciones. Por esto, hoy en día, se define la audiencia considerando lectores, radioyentes, televidentes y usuarios de internet.[13]

3.2 Objetivos de los análisis de audiencia

Existen dos objetivos principales por los que se llevan a cabo los estudios de audiencias.

1. Cubrir las necesidades de información de la industria publicitaria.
2. Aprovechar la información para que los medios planifiquen, diseñen y programen los contenidos editoriales siguiendo la respuesta de la audiencia.

Así, la medición sirve como un instrumento de diagnóstico (cuantas personas han visto determinado programa), como un instrumento de tasación (cuantas más personas hayan visto el programa, mayor posibilidad de impacto puede tener un anuncio publicitario insertado en dicho programa), como un instrumento de predicción (los resultados de audiencia pueden indicar la tendencia y resultado de futuros programas), y como instrumento de planificación (a partir de las mediciones, se suele decidir qué programas se siguen emitiendo).

3.3 Los sistemas de medición de audiencias

El interés por conocer las características de la audiencia se crea con los propios medios de comunicación. En términos generales, la medición de audiencias hace referencia al estudio para conocer el número de personas que consumen productos o se exponen a un mensaje en diferentes medios y soportes. Su finalidad es determinar el tamaño de la audiencia, clasificada a partir de diferentes variables, como el género o la edad. Partiendo de estos datos se considera que se pueden determinar tendencias sobre hábitos o comportamientos de consumo o recepción.

Una división de los estudios de audiencias clasificaría los métodos utilizados en dos grandes grupos: aquellos en los cuales la información de los individuos se obtiene por declaración y aquellos que utilizan la observación para el mismo propósito.[14] En el primer grupo están los estudios de audiencia basados en entrevistas, las encuestas, los diarios de escucha para la radio y la televisión, etc. y en el segundo los que parten de resultados recogidos de forma mecánica, como los audímetros, que proporcionan un nivel de detalle y una velocidad en la entrega de resultados superior a otras técnicas.

Otra división clasificaría los estudios según la metodología de investigación:

- Metodologías centradas en un solo medio (monográficos), en varios (multigráficos) o en medios y determinados productos (media-producto).
- Metodologías basadas en el reconocimiento. Se presentan a individuos publicaciones o espacios de televisión y radio parcial o completamente para observar si los reconocen.
- Metodología basada en el recuerdo. Se estimula la memoria de los individuos presentándoles contenidos no recientes para ver si los recuerdan.
- Metodología de la audiencia ayer por primera vez. La audiencia semanal se calcula mediante la audiencia del día anterior multiplicada por siete.
- Metodología de la frecuencia de lectura o contacto. Permite conocer la audiencia de un día y la acumulación de ésta a lo largo de un periodo de tiempo.

A continuación haremos un repaso de los principales estudios de medición de audiencias.

3.3.1 Los estudios de recuerdo

Son estudios basados en encuestas en los que los individuos explican hábitos de consumo. Son bastante caros de llevar a cabo de manera regular por lo que se evita hacerlos de forma permanente ya que se basan en la sinceridad de los encuestados.

3.3.2 Los estudios coincidentales

Se llevan a cabo justo en el momento de producirse el contacto entre el individuo y el contenido por medio de entrevistas telefónicas. Hoy en día este tipo de estudios no es muy viable como sistema de medición continuo, por su coste, las limitaciones horarias que establece y porque no incluye teléfonos móviles y por tanto audiencia en coches.

Este tipo de estudios suelen hacerse de manera complementaria a otros, de manera que las conclusiones finales se cotejan y ganan seguridad. Se utilizan mucho para contrastar los resultados del sistema de audímetros.

3.3.3 Los diarios de escucha

Es un tipo de estudio que requiere que cada uno de los individuos llevados a estudio anote cada día en un impreso su consumo de televisión para después sacar datos y conclusiones. El formulario consiste en una parrilla con las 24 horas del día divididas en fracciones de 15 minutos.

Esta técnica se creó para evitar que el individuo tenga que recurrir a recordar los que vio en un determinado momento, cosa que ocurría en las encuestas. Además, el encuestador se vuelve innecesario, lo que conlleva un ahorro en la realización del estudio. Evidentemente, este tipo de estudios también tiene desventajas. Por ejemplo, los formularios se han de entregar y recoger, lo que supone un gasto elevado. Otro de los problemas son las posibles contestaciones erróneas y la falta de respuesta de los usuarios, así como la disciplina y constancia para rellenar el informe.

3.3.4 Las encuestas de opinión

Son estudios de medición cuantitativa para el conocimiento de datos objetivos de la realidad. Es una medición estadística tomada a partir de encuestas destinadas a conocer la opinión pública. Estas mediciones se realizan por medio de muestreos que, usualmente están diseñados para representar las opiniones de una población llevando a cabo una serie de preguntas.

Se pueden llevar a cabo de manera presencial, por correo, por teléfono o por internet. Este tipo de estudios está condicionado por las preguntas (deben ser las mismas para todos los encuestados), las respuestas (pueden ser abiertas o cerradas, pueden ser dos o varias, etc.), por el entorno y por las características del individuo.

3.3.5 Los prestest y postest

Este tipo de estudio se utiliza sobre todo en Estados Unidos para el ámbito de la política y se pueden aplicar sobe un mismo producto o contenido de dos formas

complementarias. Se pueden llevar a cabo mediantes salas de proyección en los que los participantes indican si lo que ven les gusta o no, o mediantes los D-Groups, reuniones que se realizan en torno a una mesa y en las que participan personas seleccionadas por su perfil. En estas reuniones se intentas que los participantes expongan libremente sus puntos de vista.

El prestest consiste en realizar la encuesta a una pequeña muestra del público objetivo con la finalidad de detectar errores y poder subsanarlos. El postest pretende evaluar los productos con el fin de determinar el logro de los objetivos establecidos, es decir, su éxito y su fracaso, el conocimiento de las expectativas de la audiencia y la orientación e intervención en el proceso creativo.[15][16]

3.3.6 La audimetría

Este tipo de estudio consiste en la adaptación de un aparato llamado audímetro (Figura 3.1) a los televisores de los hogares de una muestra representativa. Permite medir la audiencia tanto en televisión abierta (TDT) como en televisión por cable o satélite. Se dispones de un mando en el que se incluyen tantos botones como personas residan en el hogar, de tal manera que a medida que se vayan sentando a ver la televisión cada uno pulse el botón asignado a su persona. A partir de este mando, el audímetro registra televidente, cambios de canal, cambios de volumen y mute en tiempo real. Los datos se almacenan y se transmiten todos los días mediante una llamada a un ordenador central para ser procesados.

Figura 3.1. Audímetro y control remoto

La empresa encargada del análisis de audiencia realiza estudios correspondientes para ofrecer el audímetro a un grupo estadísticamente significativo de personas. Pero la gran desventaja de los audímetros es la gran colaboración que necesitan por parte de los usuarios, lo que hace que muchos de estos se muestren reacios a instalar uno en su hogar pese a los compensaciones que se consiguen por hacerlo.

Por esta razón la industria está avanzando hacia el recuento individual pero pasivo de los espectadores. En 1957, la empresa Nielsen lo hizo con un curioso audímetro con forma de cojín que se activaba cada vez que alguien se sentaba encima. Los problemas de este modelo enseguida se hicieron patentes, ya que muchas veces en las que la gente utilizaba el cojín no estaba viendo la televisión.

En Francia se han utilizado audímetros fotoeléctricos que detectan cualquier movimiento en la habitación, incluso si el individuo está o no mirando al aparato de televisión. Modelos parecidos al anterior fueron desarrollados nuevamente por Nielsen, incorporando infrarrojos y videocámaras en lugar de células fotoeléctricas. Esto condujo a numerosas protestas por la consiguiente falta de intimidad y los modelos fueron retirados. Por estas razones, y pese a sus limitaciones, el audímetro activo-individual sigue siendo el modelo más utilizado en la actualidad.

3.3.7 La medición de audiencia por tipo de medios

Para el estudio de audiencia en los medios impresos las técnicas más frecuentes son las encuestas personales y los estudios de recuerdo para determinar la audiencia de suplementos y secciones en los periódicos. También se estudia la acumulación de lectura a lo largo de varios números, la velocidad y calidad de las lecturas y la exposición a la publicidad.

En la radio, los estudios habituales son los del recuerdo del día de ayer mediante entrevistas personales y la del diario de escucha, más caro pero más rico en información. Durante los últimos años la discusión sobre ventajas o desventajas de estos estudios ha sido prácticamente inexistente porque la atención de los profesionales ha estado concentrada en analizar la viabilidad de los audímetros aplicados a la medición de este medio.[17]

En relación con la televisión, aunque se hacen estudios de todo tipo con diferentes técnicas (de audiencia, de hábitos de medios y consumos de productos y de post-evaluación) el sistema más común es el audímetro. Más de 70 países lo utilizan para medir audiencias, estudiando el visionado de televisión de más de 250.000 personas en todo el mundo.

En cuanto a internet, la búsqueda de un sistema que consiga un grado razonable de consenso entre los compradores y vendedores de espacios publicitarios resulta casi

imposible. Los que existen son sistemas que miden las visitas a un portal, tiempo promedio, etc., a partir del simple recuento de los ficheros de los servidores de contenido. También es conocida y aplicada la técnica de medición a través de los llamadas *tags*, pequeños programas imperceptibles para el usuario que se activan cuando el navegador descarga la página y procede a enviar a través de la red un aviso de la descarga al centro de control de edición.

Por último, es interesante señalar que la audiencia de cine se ha medido a través de entrevistas a una muestra de la población ya que el control de taquilla aporta un número de personas sin proporcionar datos de perfiles (genero, edad, etc.) indispensables en la planificación de medios.

3.4 La medición de audiencia del medio exterior

Durante los últimos años se ha observado que el panorama de la publicidad exterior se ha hecho más sofisticado y complejo, donde no sólo se ha proporcionado nuevos elementos tecnológicos a las vallas tradicionales sino que el abanico de posibilidades se ha extendido a diferentes elementos del mobiliario urbano (cabinas telefónicas, papeleras, marquesinas, etc.), a recintos cerrados como el metro, los aeropuertos, los centros comerciales e incluso a la publicidad en movimiento a través del transporte público como soporte publicitario.

La técnica para medir la audiencia más habitual, la reconstrucción de las rutas seguidas por una muestra de individuos, se mantiene. Pero hay dos aspectos donde la medición se desarrolla. Por una parte, la reconstrucción de las rutas evoluciona hacia métodos donde la información se recoge de forma electrónica. Actualmente se basa en la realización de entrevistas, ya sean personales con ayuda de ordenadores y de mapas digitales o de entrevistas telefónicas que hacen uso de sistemas inteligentes para proponer la ruta más probable.

Se están proponiendo dos enfoques tecnológicos diferentes. En ambos casos, se refuerza la calidad de la recogida de información y, al no depender de la memoria, se puede ampliar el intervalo temporal cubierto para cada individuo de la muestra:

- El IPM (Individual Proximity Meter) que proponen los británicos Derek Bloom y Michael Stewart (Figura 3.2). Consiste básicamente en la colocación de unos pequeños emisores en las vallas cuyas señales, de tipo casi continuo, puedan ser

recogidos por unos dispositivos diseñados al efecto que pesan solo 53 gramos y están dotados de un sensor de movimiento y una capacidad de batería para 42 días. Estos dispositivos se entregarían a una muestra de individuos que se comprometerían a llevarlos por un período de tiempo. Cuando estos individuos pasaran a menos de 50 metros de los emisores captarían la señal particular del objeto publicitario y la archivarían en su memoria interna junto con la fecha y la hora. Tienen la ventaja de que pueden funcionar en entornos cerrados donde el seguimiento GPS falla.[18]

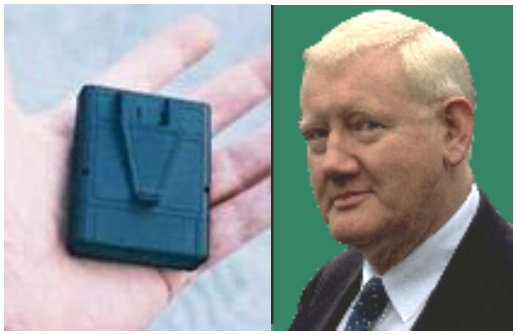

Figura 3.2. Dispositivo IPM y Michael Stewart

- La utilización de dispositivos GPS (Geographical Positioning System) por parte también de una muestra de individuos. Se han llevado a cabo experimentos en varios países como Dinamarca, Sudáfrica, Canadá, Estados Unidos, Suiza e Italia con resultados satisfactorios. Su aplicación no se percibe de la misma forma en todos los países. Así, por ejemplo, en Canadá y Estados Unidos, el dispositivo GPS se llevaría en el coche porque se entiende que una inmensa mayoría de los impactos publicitarios se produciría en el entorno del tráfico rodado, mientras que en Europa, donde la importancia de los trayectos peatonales o de utilización del transporte público es muy significativa, se tendría que prever que se pudiera llevar también en el bolsillo o en el bolso (Figura 3.4). Compañías como Nielsen y Arbitron en Estados Unidos o la italiana Eurisko están trabajando en el desarrollo y mejora de este tipo de sistemas.

Figura 3.3. GPS de la empresa EURISKO

Figura 3.4. Posibles maneras de portar el GPS

3.5 La medición de audiencias en Digital Signage

Para llevar a cabo la medición de audiencias en redes de Digital Signage, se utilizan software de medición y detección de audiencias. Dicho software detecta los rostros de la audiencia mediante sensores ópticos y los relaciona con una ID suministrada por el propio software. A esta ID es a la que se relaciona la información demográfica de la audiencia, como la edad, el género, etc. e información del tiempo que se encuentra delante del sensor óptico. De esta manera el estudio de la audiencia es totalmente anónimo.

En el siguiente capítulo se verán distintos software de medición y detección de audiencias.

Capítulo 4. Software de medición de audiencia

En este capítulo se verá información sobre diferentes software de medición y detección de audiencias, método utilizado en la medición de audiencias en Digital Signage, dando mayor relevancia a los usados en este Trabajo de Fin de Grado.

4.1 SOFTWARE DE MEDICIÓN DE AUDIENCIAS

A medida que avanza la tecnología los consumidores se ven cada vez más bombardeados con anuncios en cualquier parte, por otra parte los empresarios necesitan saber si estos anuncios están cumpliendo su función o por el contrario no están siendo vistos por nadie, ya sea por la localización o por la falta de interés de los consumidores. Para solventar estas dudas, los anunciantes están optando por los letreros digitales que ofrecen una experiencia significativa respecto a los anteriores y permiten a los empresarios una promoción más eficaz.

Existen diferentes tipos de software que permiten medir la audiencia en tiempo real a través de, por ejemplo, una cámara web. Este software monitorea las métricas de la audiencia, como el género, el rango de edad al que pertenece y el tiempo de atención. Con estos datos, los anunciantes puede presentar el anuncio que esté más acorde con la audiencia.

Vamos a ver varias empresas que disponen del software mencionado.

4.1.1 Intel AIM

El Intel Audience Impression Metric Suite (Intel AIM Suite) es un sistema de medición de audiencias completamente automatizado que utiliza tecnologías Anonymous Viewer Analytics (AVA). Intel AIM Suite está impulsado por la tecnología de detección de rostros llamado Intel AIM Suite Audience Counter. Audience Counter es una tecnología basada en software que utiliza una cámara para recopilar las métricas de audiencia para la señalización. Las métricas registradas incluyen la audiencia total, los tiempos de visualización, y la información demográfica como el género y el rango de edad.

Por razones de privacidad, todos los datos recogidos son anónimos y no pueden asociarse a un individuo específico. Estas métricas se pueden utilizar para medir la efectividad de las campañas publicitarias, para adaptar el contenido de pantalla basado en características de la audiencia, y para determinar las mejores ubicaciones para la señalización.[19] En el apartado 4.2 se ampliará la información.

4.1.2 Quividi

Quividi es una empresa francesa creada en julio de 2006. Esta empresa dispone de una solución para la medición y detección de audiencia muy usada en todo el mundo. Mediante el uso de un sencillo sensor óptico, Quividi emplea una tecnología propia patentada para contar los espectadores actuales así como las OTS (Oportunity to See), al mismo tiempo que mide con precisión los tiempos de exposición y los tiempos de atención así como una estimación de la distribución por género y por grupo de edad.

La misma tecnología puede emplearse como contador de paso, para conocer la afluencia en los lugares clave.

Este conjunto de métricas están disponibles en tiempo real, de forma simultánea, los datos se envían a la nube, de forma que permitirá hacer un análisis de la efectividad de su comunicación, sabiendo el grado de éxito y permitiendo mejorar el retorno de la inversión (ROI).[20] En el apartado 4.3 se ampliará la información.

4.1.3 GEOMEX

GEOMEX es un sistema de medición de audiencia Out-of-Home establecido en España. Gracias a GEOMEX, es posible disponer de los datos de audiencia de las redes de publicidad exteriores en las principales ciudades españolas. Este sistema de medida está actualmente en uso en otros países.

Los datos de audiencia se pueden integrar con el software desarrollado por CUENDE Infometrics, así como software de planificación de otras empresas como Tom Micro. Además, CUENDE dispone de datos de audiencia para publicidad en el metro mediante GEOMETRO o de audiencia en de la publicidad en los autobuses de áreas metropolitanas mediante GEOTRANS.[21]

4.1.4 Crystal Displays

Crystal Displays nos permite capturar la edad de la audiencia, el género, medir el número de impresiones, medir el dwell time (tiempo de permanencia) y medir el flujo de compradores en las tiendas. Ha sido desarrollado específicamente para su uso en sistemas de señalización digital. Usa la detección de rostros y la tecnología de seguimiento para detectar las caras y las impresiones captadas por dispositivos ópticos digitales.[22]

Crystal Displays Analytics proporciona inteligencia que le permite comprender las características de la audiencia capturada por las pantallas, incluyendo:

- Impresiones reales. El número de personas que mira las pantallas.
- Duración. Cuanto tiempo está mirando a pantalla
- Tamaño de la audiencia potencial. El número de personas que pasan por delante del dispositivo óptico sin mirar a la pantalla.
- Dwell Time. Cuanto tiempo su queda cerca de la pantalla.
- Demografía. Género y edad de la audiencia.

Utilizando estos parámetros se pueden obtener beneficios, incluyendo:

- Entender ROI
- Entender ROO (Return on Objectives) tales como cambios en el comportamiento del consumidor
- La determinación de las mejores ubicaciones para pantallas
- Adaptación de contenido de la pantalla en base a características de la audiencia
- Optimización de la publicidad basada en los datos de medición de audiencia exacta

4.1.5 TruMedia

iCapture mide con precisión la atención del público delante de las pantallas. Sirve tanto en tiempo real como en la medición de la audiencia a través de la TruMedia Dashboard. iCapture mide el comportamiento de los compradores cerca de los monitores y de las pantallas. Se compone de una smartbox (pequeña caja de procesamiento de vídeo) y uno o varios sensores (Figura 4.1). iCapture detecta y rastrea automáticamente rostros captados por los sensores.

Mientras iCapture Solo puede medir la atención visual de una pantalla, iCapture Duo tiene la flexibilidad de la medición de dos pantallas separadas en diferentes lugares, por tanto, optimizar la funcionalidad.[23]

Figura 4.1. Smartbox y sensor de iCapture

4.1.6 relEYEble

relEYEble es una herramienta de medición y análisis de audiencias que permite medir el impacto real de las campañas de marketing: género, rango de edad, impresiones y la duración de éstas, así como otros datos pertinentes. Permite interactuar con el público y presentar el contenido personalizado de acuerdo a la audiencia real (sexo y rango de edad) en tiempo real y optimizar la eficacia de sus campañas de marketing mediante la adición de interactividad a sus redes de señalización digital.[24]

Figura 4.2. Gráficas ejemplo de relEYEble

4.2 PLATAFORMA INTEL AIM SUITE

Las características de Intel AIM Suite permiten a los anunciantes maximizar el retorno de la inversión de sus campañas. Esta solución (Figura 4.3) comprende cuatro elementos:

- Intel AIM Suite.
- Intel AIM View.
- Intel AIM Analytics.
- Intel AIM Manage.

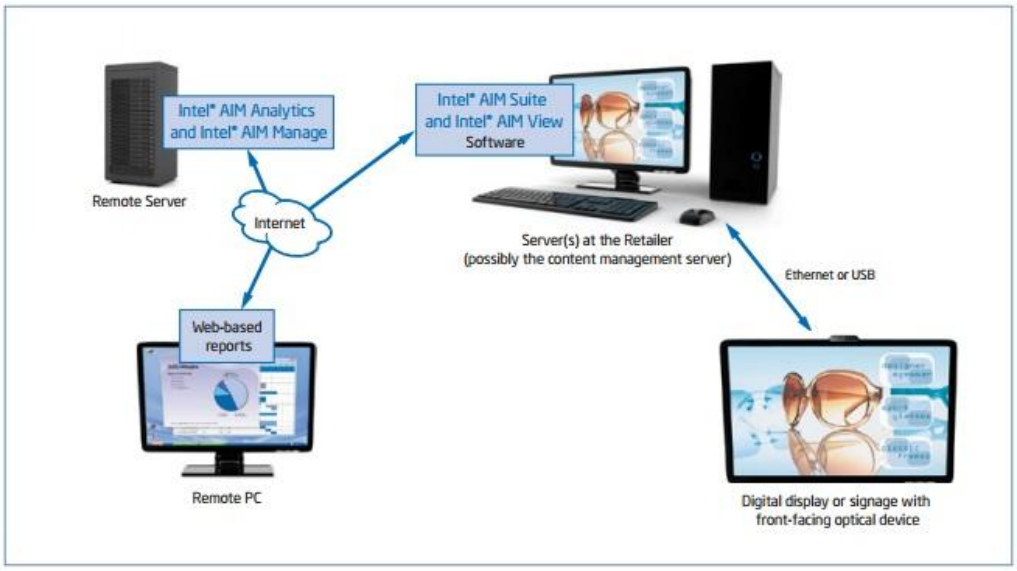

Figura 4.3. Ejemplo de configuración de una red usando Intel AIM Suite

4.2.1 Intel AIM Suite

Se trata de la aplicación de control para la solución AVA. Gestiona las peticiones de Intel AIM View y envía los datos a Intel AIM Analytics. Este módulo se instala en el ordenador e incluye el módulo Intel AIM View (Figura 4.4).

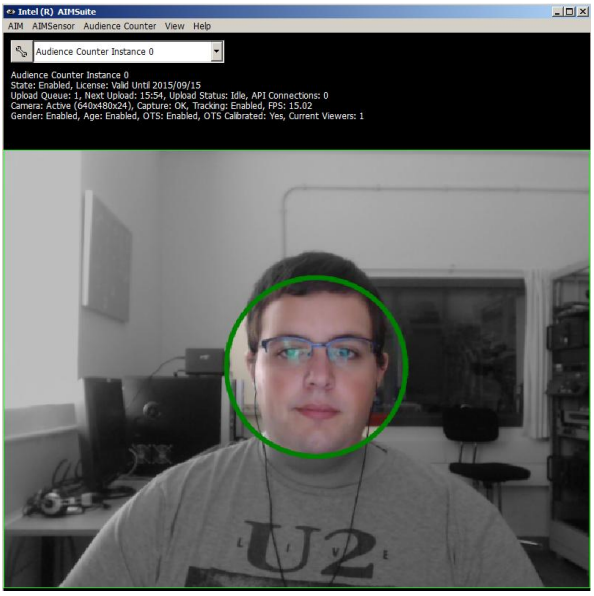

Figura 4.4. Captura de Intel AIM Suite, elaboración propia

Desde Intel AIM Suite se configura en sensor óptico y la pantalla de captura, desde configurar las distancias mínima y máxima de medida hasta crear una máscara para indicar que en esa zona no tome valores.

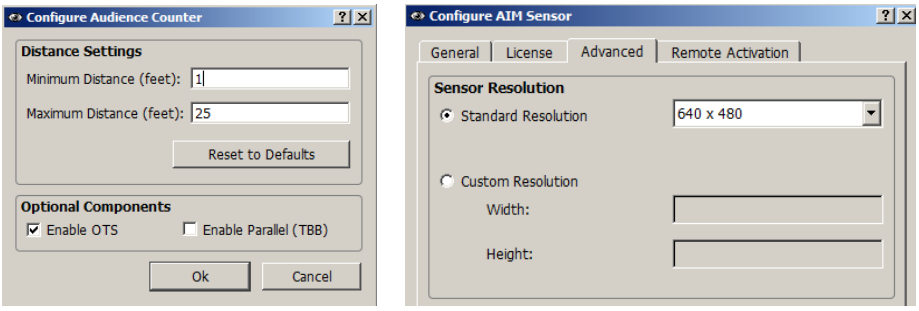

Figura 4.5. Ventanas de configuración en Intel AIM Suite, elaboración propia

En este módulo se configurará el sensor óptico la primera vez, introduciendo la licencia que se nos proporcionará en Intel AIM Manage. Intel AIM Suite nos proporcionará, además de una visión de lo que el sensor capta, información sobre la configuración del Intel AIM View y que información se está enviando al Intel AIM Analytics, como fecha de caducidad de la licencia, si captura el género, la edad, las OTS, etc. (Figura 4.6). Una vez instalado Intel AIM Suite, deberá estar siempre ejecutándose ya que, como se comentó anteriormente, es el módulo encargado de gestionar las peticiones de Intel AIM View. Una vez que se configure tanto el sensor como la pantalla de captura, el módulo pasará a ejecutarse en segundo plano pudiendo acceder a él desde la esquina derecha de la barra de tareas (Figura 4.7).

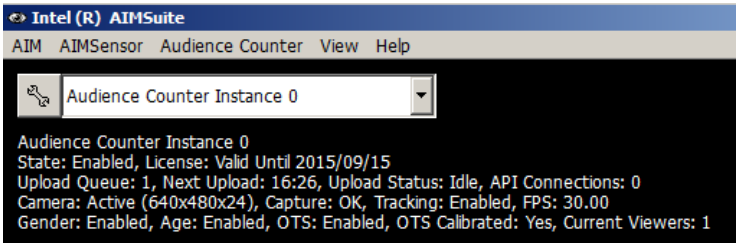

Figura 4.6. Captura de la información, elaboración propia.

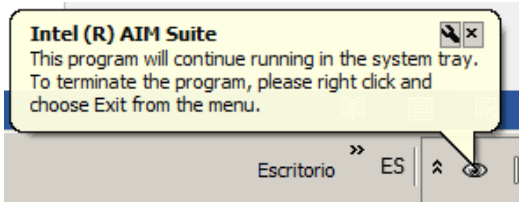

Figura 4.7. Captura de la situación del icono en la barra de tareas, elaboración propia

4.2.2 Intel AIM View

Intel AIM View es el módulo de detección de rostros de Intel AIM Suite. El software analiza una secuencia de video desde una cámara y detecta los rostros de la audiencia, proporciona información sobre el número de espectadores, su género, su rango de edad y el tiempo de visualización.

Figura 4.8. Ejemplo de funcionamiento de Intel AIM View

4.2.3 Intel AIM Analytics

El Intel AIM Analytics se trata de un sistema de información basado en la web. Desde la página web de Intel AIM podemos acceder al Analytics y observar los datos recogidos de forma gráfica, ya sea en gráficas preestablecidas por Intel o gráficas que el usuario quiera crear. Permite exportar los datos deseados en un texto csv (comma-separated values) que se pueden abrir desde Microsoft Excel y ver los datos en formato tabla.

Figura 4.9. Captura de gráficas preestablecidas por Intel, elaboración propia.

4.2.4 Intel AIM Manage

Intel AIM Manage es el sistema de gestión de sensores de Intel AIM Suite. Este sistema basado en la nube gestiona de forma remota todos los equipos con Intel AIM Suite. Desde la página web de Intel se podrá acceder al Manage y ver los sensores configurados en ese momento o añadir otros (Figura 4.10). Aquí se deberá acceder para descargar el ejecutable de Intel AIM Suite que comentábamos en el apartado 4.2.1. También se permite el acceso a archivos necesarios para la configuración de la API.

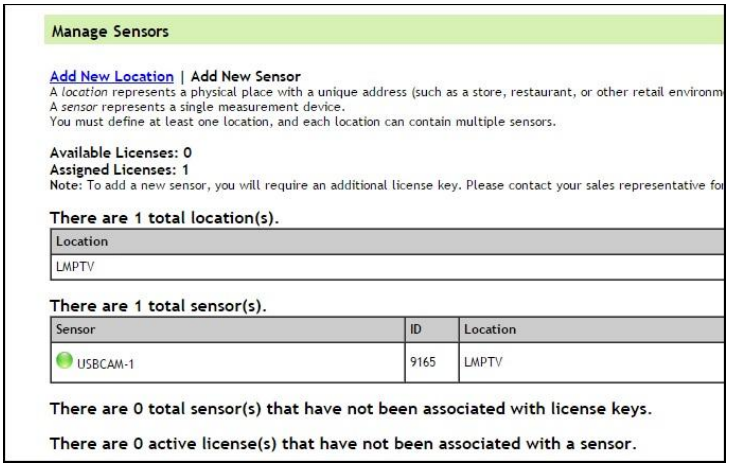

Figura 4.10. Captura de Intel AIM Manage, elaboración propia

4.3 PLATAFORMA QUIVIDI

Quividi, al igual que Intel AIM Suite, necesita un programa que debe ejecutarse en el ordenador en todo momento. La solución de Quividi para la medición de la audiencia incluye dos módulos:

- VidiReports
- VidiCenter

4.3.1 VidiReports

VidiReports analiza el flujo de video proporcionado por un sensor óptico para ofrecer una gran variedad de métricas de audiencia en tiempo real en tantas pantallas como sensores ópticos tengamos, se trata de un módulo basado en la web. Quividi dispone de dos modos de funcionamiento, VidiReports o Vidigates. VidiReports se encarga de la medición de audiencia tal y como se utilizará en este TFG (Figura 4.11) mientras que VidiGates se encargaría de medir el flujo de personas que pasan a través de una puerta o zona de estudio (Figura 4.12). Evidentemente según el modo de funcionamiento se deberá modificar la colocación del sensor óptico, para el caso de VidiReports el sensor óptico deberá colocarse en el frontal de la pantalla de estudio, mientras que para VidiGates el sensor óptico deberá estar dirigido hacia el suelo de la zona de estudio.

Figura 4.11. Colocación del sensor óptico para VidiReports

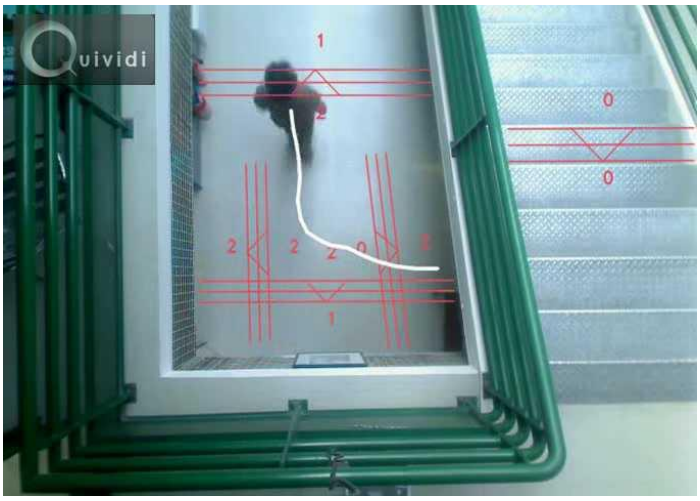

Figura 4.12. Colocación del sensor óptico para VidiGates

Entrando en la página proporcionada por Quividi se podrá configurar el sensor óptico, la frecuencia en la que los datos se cargarán en el servidor, el modo de funcionamiento (VidiReports o VidiGates) o crear máscaras para la pantalla de captura.

4.3.2 VidiCenter

VidiCenter es la solución diseñada como complemento para VidiReports y VidiGates. VidiCenter reúne de forma segura las métricas de la audiencia de todas las unidades con VidiReports o VidiGates y automáticamente construye un panel gráfico informativo accesible a través de un simple navegador web

Con VidiCenter se puede seguir en tiempo real todos los datos de la audiencia de las redes de DS, para el caso de VidiReports, datos como el número de la audiencia total, las OTS, etc. Se podrá seguir la evolución de la audiencia y comparar el rendimiento de cada lugar viendo los datos en gráficas preestablecidas por Quividi o en gráficas creadas por el usuario.

En el portal Vidicenter se configuraran los sensores ópticos. Existe la opción de agrupar los sensores en localizaciones o en redes de DS, de esta forma se podrá nombrar una red como ULPGC-Campus Tafira y agrupar en dicha red los distintos sensores ópticos que se hayan desplegados por el campus.

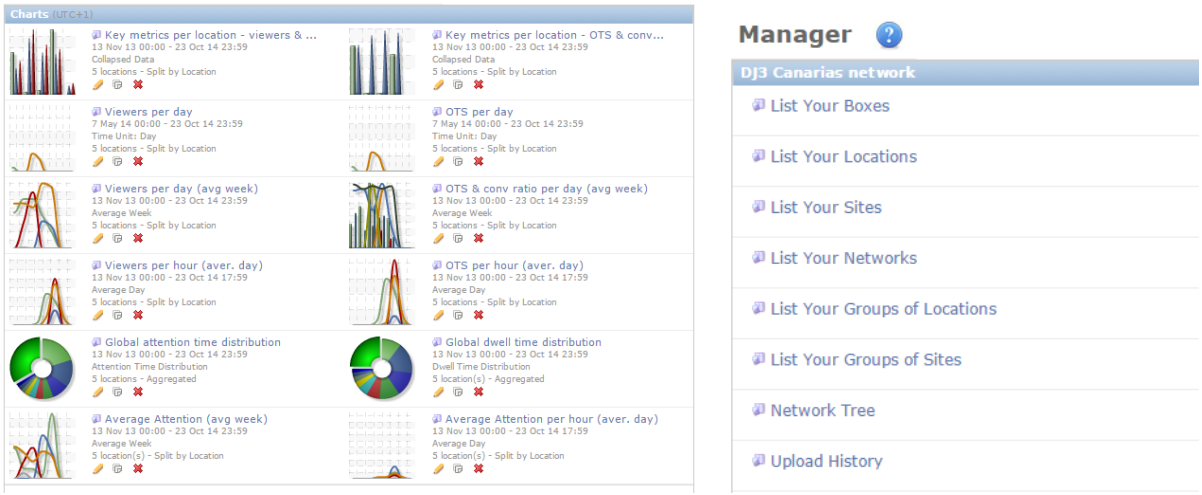

Figura 4.13. Capturas de VidiCenter, elaboración propia

Capítulo 5. SCALA

En este capítulo se estudiará en el gestor de contenidos Scala, gestor de contenidos que ha sido utilizado para la realización de pruebas en este Trabajo Fin de Grado.

5.1 Opciones de Software

Scala ofrece dos opciones para trabajar con su software:

- Enterprise. Esta opción nos ofrece un paquete completo que incluye el Content Manager, Designer y el Player de Scala.

- Scala as a Service. Es una versión de Scala Content Manager instalada en la red que ofrece la gestión en línea de la red de señalización digital. No requiere instalación de software ni hardware para el servidor. Este servicio en línea, al igual que otros modelos de SaaS, permiten acceder al software de Scala. El Content Manager únicamente se ve y se accede por la empresa que contrate el servicio. Scala se encarga de todo el trabajo necesario en el servidor, incluyendo la implementación, configuración, monitorización, actualizaciones, copias de seguridad, mantenimiento de hardware, y la recuperación del sistema. También incluye Scala Certified Partner, un servicio que nos ayuda a diseñar la red de DS.

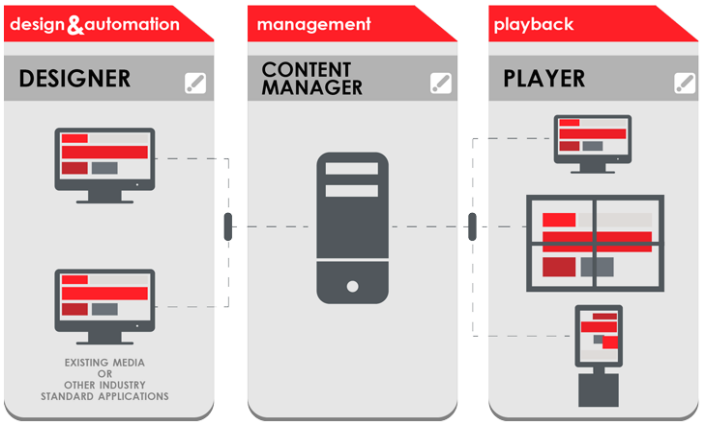

Figura 5.1. Distribución de herramientas de Scala

5.2 Designer

Scala Designer es la solución más rápida para la creación de contenidos verdaderamente dinámicos para la DS. El Designer maneja una gran variedad de formatos de imagen (BMP, JPG, GIF, PNG, TIFF); video (AVI, SWF, MPG, WMV, H.264); y audio (WAV, MP3), formatos que podremos agregar a nuestros diseños. Dispone de un complemento de Photoshop que permite crear un boceto en Photoshop y luego exportarlo a Designer en donde se podrá agregar tiempos, transiciones, etc. Permite combinar texto, gráficos, sonido y video y realizar una vista previa de exactamente tal y como se verá el diseño.

Designer permite sacar partido de cientos de efectos y transiciones integrados en el motor de reproducción Scala. Nos permite añadir botones y variables para diseñar un contenido interactivo. Es compatible con varios lenguajes de programación, como ScalaScript, VBScript, JavaScript o Python. Dispone de complementos llamados complementos de EXtensión, como por ejemplo:

- **Módulo Sintonizador de TV (TV Tuner)**: Incorpora programas de TV a la pantalla, o en una ventana.
- **Módulo CORIOgen**: Convierte una señal de VGA en una señal de TV que aparecerá en la pantalla.
- **Módulo del Estado de Tiempo (Weather)**: Convierte la pantalla en un centro de meteorológico con datos en tiempo real del estado del tiempo desde una estación climatológica.
- **Módulo de control de video**: Cambie entre transmisiones de video con conmutadores controlados mediante RS-232.

Al terminar de diseñar, Designer nos permite publicar el contenido en Content Manager en donde se programa y se distribuye a los reproductores Scala Player.

Figura 5.2. Captura de Scala Designer, elaboración propia.

5.3 Content Manager

El sistema Content Manager de Scala permite mejorar y simplificar la administración y control de la red de DS. Al tratarse de un servicio basado en la web podemos mantener el control de nuestra red desde cualquier parte del mundo, desde cualquier ordenador conectado a internet.

Dispone de plantillas de contenido que facilitan la creación rápida de pantallas de gran impacto visual, nos permite organizar el contenido para que todo sea fácil de encontrar y crear listas de reproducción usando guiones, mensajes, videos, flash o imágenes estáticas de Scala.

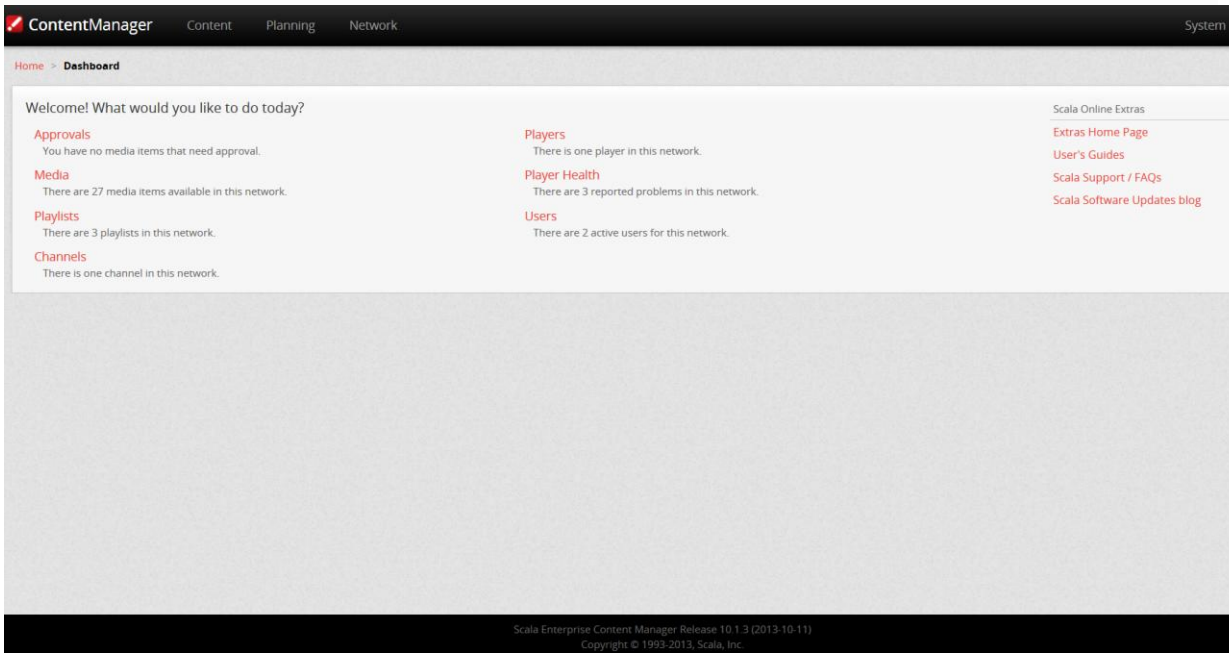

Figura 5.3. Captura de la pantalla de inicio del Content Manager

Desde Content Manager se puede definir y editar horarios para las listas de reproducción, con patrones de recurrencia variable y fechas de inicio/final; configurar mensajes para que tengan prioridad sobre las pantallas a horas específicas o cuando ocurran eventos externos, como por ejemplo emergencias; usar metadatos del reproductor para personalizar el contenido de manera dinámica; crear esquemas múltiples dentro de una pantalla en la que cada esquema reproduce contenido independiente; asignar dos canales separados a un solo reproductor, reduciendo las necesidades de hardware y licencias de sistemas operativos; crear canales de solo audio o reproducir música que se puede ignorar cuando un video tenga sus propias pistas de audio; agrupar reproductores por geografía, demografía, etc., configurar opciones de reproducción y seleccionar el contenido

a reproducir; revisar el estado de la red mediante el monitoreo del estado del reproductor; realizar tareas de mantenimiento como reiniciar, enviar y recuperar archivos o instalar actualizaciones de Scala a distancia, estas tareas se pueden programar para que se ejecuten en momentos que no interrumpan el buen funcionamiento de la red de DS.

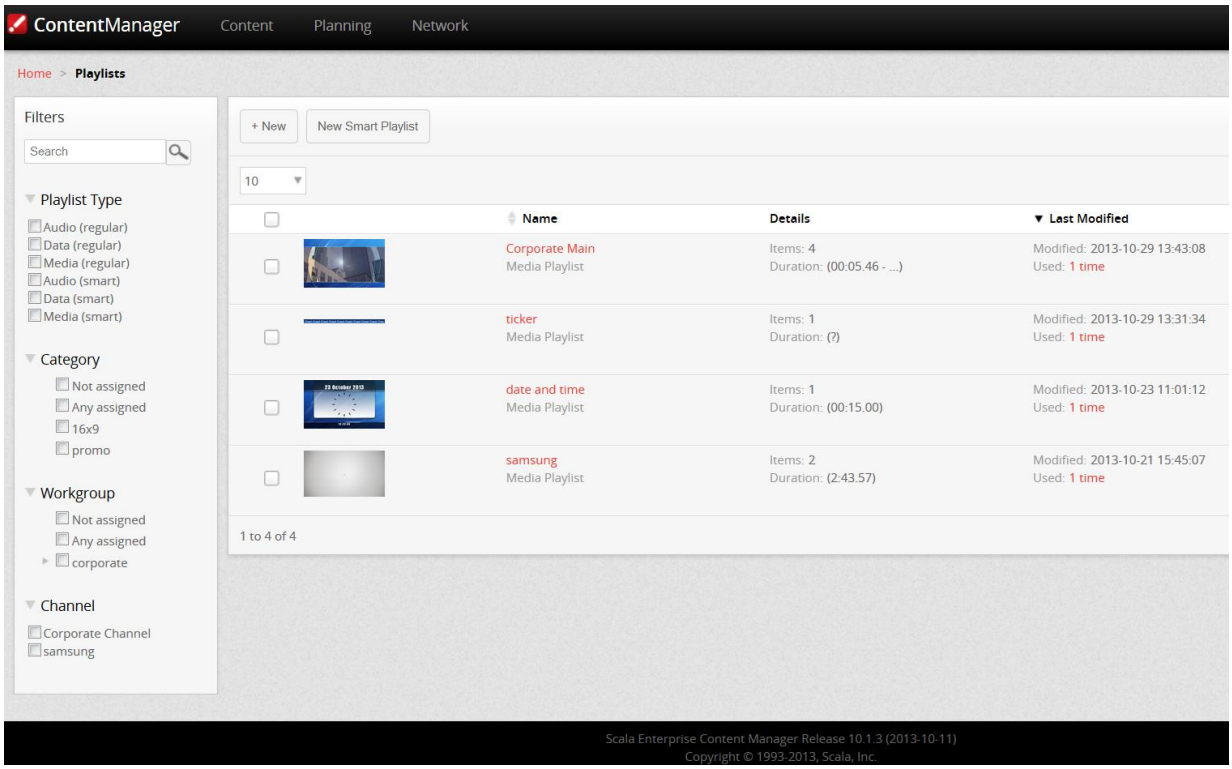

Figura 5.4. Captura de Playlists en Content Manager

Es compatible con proxies y con ACNS (Application Content Networking System), y puede entregar contenido desde HTTP/HTTPS mediante IPv4 o IPv6 con lo que se integran la mayoría de los requisitos de seguridad de tecnología de la información. Permite trabajar con el servidor y la base de datos que mejor funciones en nuestro entorno, entre las bases de datos tenemos opciones de usar se incluyen MySQL, Microsoft SQL Server y PostgreSQL.

Content Manager dispone de distribución inteligente de contenido, es decir, el contenido se transmite automáticamente al permitir que cada reproductor selecciones de forma inteligente solamente los archivos que son nuevos o que hayan cambiado, disminuyendo así el número total de archivos a descargar.

El Content Manager dispone, al igual que el Designer, de complementos extras llamados complementos de EXtensión con los que se incrementa nuestra capacidad de manejar el contenido de nuestra red de DS:

- **Módulo para crear plantillas (Template Composer):** cualquiera puede crear pantallas fascinantes una vez que los diseñadores gráficos de su empresa hayan establecido la “apariencia y sensación” usando el sistema Template Composer para preparar formularios que otros puedan llenar.
- **Módulo de Auditoría de Reproducción (Playback Audit):** Playback Audit procesa los registros de reproducción de cada Scala Player para posteriormente generar informes sobre el contenido reproducido.
- **Módulo de Servidor de Transmisiones (Broadcast Server):** Este módulo permite enviar contenido a través de un satélite y redes de transmisión, distribuyendo el contenido a cientos de reproductores con una sola transmisión. Esto será útil cuando nuestro número de reproductores sea muy grande.

5.4 Player

Scala proporciona el Scala Player. Dispone de una tecnología de reproducción que entrega movimiento a nivel sub-píxel sin saltos. El Player se comunica con el Content Manager para vigilar el estado general del reproductor Player y registrar los eventos para facturación de anuncios y otros reportes.

Windows Script permite una interfaz entre el sistema de reproducción Player y otros sistemas que usen JavaScript, Python o VBScript. Soporta Windows Streaming o entradas de video en vivo mediante el complemento con un módulo de EXtension y su correspondiente hardware. Reproduce tanto contenido cargado desde Content Manager como contenido interactivo mediante una pantalla táctil. Tenemos la opción de instalarlo en pantallas anchas o auto rotación en pantallas portátiles. Si la conexión de red se cae, el sistema de reproducción Player sigue operando usando tecnología de almacenamiento y reenvío (store and forward).

Al igual que el Designer o el Content Manager, el Player dispone de módulos de EXtension. El Player comparte muchos módulos con el Designer, además de los mencionados anteriormente existen los siguientes:

- **Módulo de colas de espera (Queue)**: mantenga la línea moviéndose mediante la integración de un sistema de colas dentro de su red.
- **Módulo de verificación local**: Verifique el software de su sistema Player sin conectarlo al Internet.

Figura 5.5. Captura de Scala Player, elaboración propia.

Existen opciones de licencia del sistema de reproducción Player para una amplia variedad de hardware, desde lo muy básico hasta lo más avanzado. Las pantallas son más que una herramienta para compartir el contenido, es la pieza final en las redes de DS. Existen muchas opciones disponibles, cada una con diferentes características para diferentes situaciones. Scala ofrece equipos según las características deseadas.

- **IAdea Photo Frames**. El sistema IAdea Frames opera sobre una red Scala que se conecta a un Content Manager, con lo que conseguimos una gran administración de contenido y hardware de bajo costo con una pantalla integrada.
- **Equipos IAdea de definición estándar**. Los equipos de medios IAdea ofrecen servicios de video, imágenes fijas y sonido, todo en una configuración incorporada en un equipo. Existen disponibles varios modelos, incluyendo versiones en definición estándar (calidad DVD) y en alta definición.
- **Equipos IAdea de alta definición**. Los equipos IAdea HD tienen las mismas funciones de los equipos de definición estándar con la capacidad adicional de poder ejecutar contenido de alta definición.
- **Ordenadores de bajo coste**. Las mejoras en el desempeño del sistema Scala han hecho que el software sea compatible con equipos más baratos.

- **Ordenadores**. El modelo para PC de Scala cumplirá y superará sus expectativas de cartelería digital.

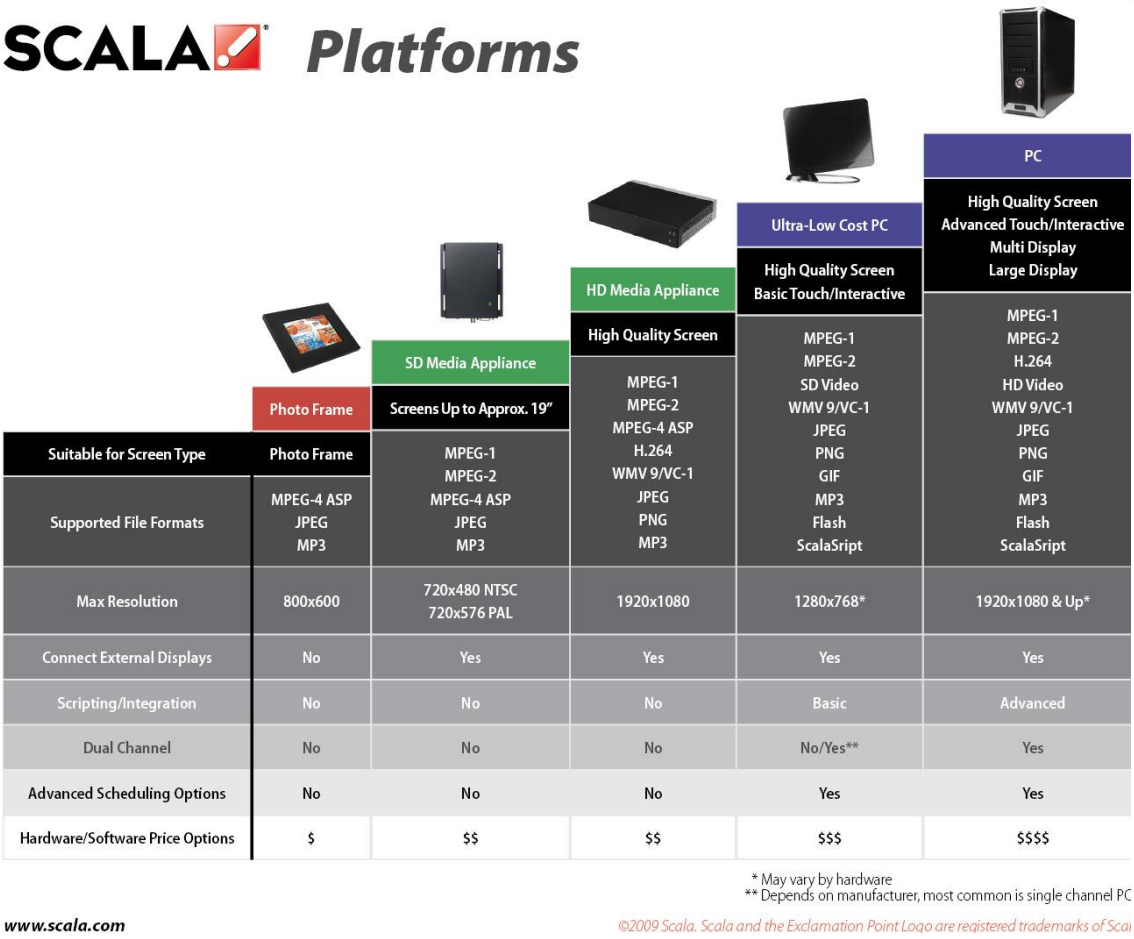

Figura 5.6. Opciones de plataformas

Capítulo 6. Desarrollo del módulo software

En este capítulo se verá el desarrollo llevado a cabo para la creación del módulo software de interacción.

6.1 Módulos software

En este Trabajo Fin de Grado se han utilizado dos softwares de medición y detección de audiencias por lo que se han creado dos módulos software de interacción. A continuación se verá los pasos llevados a cabo para el desarrollo de ambos módulos.

6.2 QUIVIDI

La empresa Quividi proporciona un SDK bastante básico en distintos lenguajes de programación. En un primer momento se eligió desarrollar el módulo de interacción en Java, pero el SDK proporcionado tenía errores a la hora de sincronizar con el servidor. Estos problemas llevaron a la prolongación de este Trabajo Fin de Grado debido a los intentos infructuosos para solventarlos. Tras el estudio de la herramienta Scala, se advirtió que dicha herramienta no era compatible con Java. Estos dos hechos hicieron que finalmente el lenguaje de programación elegido fuese Python.

En primer lugar se creó un archivo Python cuya ejecución crease un archivo de texto llamado *log.txt* el cual serviría como registro de la audiencia. El código es el siguiente:

```
#!/usr/bin/env python
#
# Importamos los módulos que usaremos
#
import sys
import time
import os.path

#Comprueba si existe la carpeta
if not os.path.isdir('W:\AudienceLogs'):
   #Si no existe la crea, crea el archivo y escribe el encabezado
   os.mkdir('W:\AudienceLogs')
   archi=open('W:\AudienceLogs\log.txt','a',0)
   time.sleep(1)
   archi.write("IDWatcher, Fecha, Hora, Tiempo, Tiempo Atencion, Genero, Rango de Edad, Contenido, Numero de Hombres, Numero de Mujeres\n")
else:
   #Si existe la carpeta, crea o abre el archivo y escribe el encabezado
   archi=open('W:\AudienceLogs\log.txt','a',0)
   archi.write("IDWatcher, Fecha, Hora, Tiempo, Tiempo Atencion, Genero, Rango de Edad, Contenido, Numero de Hombres, Numero de Mujeres\n")
```

Posteriormente, se desarrolló el módulo de interacción entre la herramienta Quividi y el gestor de contenidos Scala. Este módulo se puede dividir en tres partes; conexión con el servidor, toma de datos y gestión de datos. Existe además un código inicial en el cual se declaran variables necesarias para el módulo y pequeñas funciones como la escritura en el archivo de texto o las necesarias para tomar el valor de la edad y el género en formato texto. Este código inicial es el siguiente:

```
#!/usr/bin/env python
#
#Importamos los modulos que usaremos
#
import sys
import datetime
import time
import socket
import struct
from scala5 import *

# Recuperamos las variabels compartidas de Scala
scalavar = sharedvars()

#
# Constantes de sistema
#
MSG_TYPE_LIST = { 'EVENT_CONFIGURE_MSG' : 0x20,
                  'WATCHER_EVENT_MSG'   : 0x30,
                  'MOTION_EVENT_MSG'    : 0x40,
                  'OTS_COUNT_MSG'       : 0x60}

MAGIC_WORD          = 0xCAFE
MAGIC_WORD_STR      = '\xFE\xCA'
PROTOCOL_VER        = 0x02
MAX_PAYLOAD_SIZE    = 1024

#Formato de los mensajes
HEADER_PACK_FORMAT  = '=HBBHH'
HEADER_SIZE         = struct.calcsize(HEADER_PACK_FORMAT)

MSG_PACK_FORMAT     = { 0x20 : '=BBxx',
                        0x30 : '=LLLBBxxxxxxLLHBB',
                        0x40 : '=HHHHHLLBxxxB',
                        0x60 : '=LLLHB',
                       }
#Distintos modos de funcionamiento
EVENT_MODE_ID_LIST  = { 'NULL MODE'    : 0x00,
                        'WATCHER MODE' : 0x01,
```

```
                    'TRACKED MODE' : 0x02,
                    'MOTION MODE'  : 0x03,
                    'OTS_ONLY_MODE': 0x04,
                   }
#Estados de la audiencia
STATUS_TO_STR = { 0: '0x00: Not yet Validated watcher',
                  1: '0x01: Not Validated watcher is Dead',
                  2: '0x02: Validated wacther',
                  3: '0x03: Validated watcher is Dead'
                 }
STATUS_MASK = 0x1F

IS_WATCHING_STATUS = 0x20

MOTION_STATUS_TO_STR = { 0x00: '0x00: Not yet Validated watcher',
                         0x01: '0x01: Not Validated watcher is Dead',
                         0x02: '0x02: Validated wacther',
                         0x03: '0x03: Validated watcher is Dead',
                         0x10: '0x10: Selected Not Validated watcher is Dead',
                         0x11: '0x11: Selected Not Validated watcher is Dead',
                         0x12: '0x12: Selected Validated watcher',
                         0x13: '0x13: Selected Validated watcher is Dead'
                        }

#Funcion que devuelve el genero y la edad
GENDER_TO_STR = { 0: 'Desconocido',
                  1: 'Hombre',
                  2: 'Mujer',
                 }

AGE_TO_STR = { 0: 'Desconocido',
               1: 'Joven',
               2: 'Joven Adulto',
                            3: 'Adulto',
                            4: 'Senior',
             }

#Rutina que escribe en el archivo log.txt
def grabartxt():
   archi=open('W:\AudienceLogs\log.txt','a',0)
   archi.write(linea)
   archi.close()
```

Para la conexión al servidor se ha utilizado un socket, a través de éste se envían mensajes de configuración y se reciben datos sobre la audiencia.

```
#Configuracion del socket
SERVER_HOSTNAME = '127.0.0.1'
SERVER_PORT     = 1974
if len(sys.argv) > 1:
  add = sys.argv[1].split(':')
  if len(add) > 0:
    SERVER_HOSTNAME = add[0]
  if len(add) > 1:
    SERVER_PORT = int(add[1])
MAX_CNX_RETRY   = 6
RETRY_DELAY     = 4

#Especificamos el tipo de mensaje que queremos recibir, las frases que empiezan con
# # no las lee el sistema
EVENT_MODE      = 'TRACKED MODE'
#EVENT_MODE      = 'WATCHER MODE'
#EVENT_MODE      = 'MOTION MODE'
#EVENT_MODE      = 'OTS_ONLY_MODE'

# 1/ Conectamos al servidor
sck = None
tries = 0
while sck is None and tries < MAX_CNX_RETRY:
    errorMsg = None
    try:
        sck = socket.socket(socket.AF_INET, socket.SOCK_STREAM)
        sck.connect((SERVER_HOSTNAME, SERVER_PORT))
    except socket.error, msg:
        sck = None
        errorMsg = "Unable to connect, Socket error: '%s'\nRetrying in %d sec.\n" % (msg,
RETRY_DELAY)
    except Exception, e:
        sck = None
        errorMsg = "Unable to connect, Exception: '%s'\nRetrying in %d sec.\n" % (str(e),
RETRY_DELAY)
    if errorMsg is not None :
        tries += 1
        sys.stderr.write(errorMsg)
        time.sleep(RETRY_DELAY)
if sck is None:
    sys.stderr.write("Unable to connect to %s:%d\nGiving up !\n" % (SERVER_HOSTNAME,
SERVER_PORT))
    sys.exit(-1)
sys.stdout.write("Connected to %s:%d\n" % (SERVER_HOSTNAME, SERVER_PORT))
```

```
# 2/ Mandamos el mensaje de configuracion
msgType = MSG_TYPE_LIST['EVENT_CONFIGURE_MSG']
packet = struct.pack( HEADER_PACK_FORMAT,
                MAGIC_WORD,
                PROTOCOL_VER,
                msgType,
                0,
                struct.calcsize(MSG_PACK_FORMAT[msgType]) )
packet += struct.pack(MSG_PACK_FORMAT[msgType],
                EVENT_MODE_ID_LIST[EVENT_MODE],
                0 )
try:
   sck.sendall(packet)
except Exception, e:
   sck.close()
   sys.exit(-2)
```

Una vez conectado al servidor y enviado los mensajes de configuración, se reciben datos de la audiencia. Se recibirá un mensaje que contendrá toda la información, luego se deben asignar variables a los datos para facilitar la gestión de éstos.

```
#3/ Esperamos los mensajes de respuesta del servidor
#Iniciamos las variables
masc=0
fem=0
mascID=[]
femID=[]
try:
   while 1:
      # Comprobamos la MagicWord
      headerData = ''
      while headerData != MAGIC_WORD_STR:
         headerData += sck.recv(1)
         if len(headerData) == len(MAGIC_WORD_STR) and headerData != MAGIC_WORD_STR:
            headerData = headerData[1:]
      #Leemos la cabecera del mensaje
      headerData += sck.recv(HEADER_SIZE-len(MAGIC_WORD_STR))
      # Descomprimimos el mensaje
      (rcvMagicWord, rcvVersion, rcvMsgType, rcvReserved, rcvPayloadSize) \
      = struct.unpack(HEADER_PACK_FORMAT, headerData)
      # Comprobamos el payload
      if rcvPayloadSize > MAX_PAYLOAD_SIZE:
         continue
      if rcvPayloadSize > 0 or rcvVersion != PROTOCOL_VER:
         msgData = sck.recv(rcvPayloadSize)
      #Tomamos la fecha y hora del sistema
      ts = datetime.datetime.now()
```

```
    ts = "%04d-%02d-%02d %02d:%02d.%02d" % (ts.year, ts.month, ts.day, ts.hour,
ts.minute, ts.second)
    # Comprobamos si es WATCHER_EVENT_MSG
    if rcvMsgType == MSG_TYPE_LIST['WATCHER_EVENT_MSG']:
      sys.stdout.write("[%s] WATCHER_EVENT_MSG received\n" % ts)
      l                                                                    =
struct.calcsize(MSG_PACK_FORMAT[MSG_TYPE_LIST['WATCHER_EVENT_MSG']])
      if len(msgData) >= l:
        # Descomprimimos el mensaje
        ( startTime,
          duration,
          attentionTime,
          gender,
              age,
          watcherID,
          status,
          estimatedDistance,
          numGlances,
          notUsed)\
          =
struct.unpack(MSG_PACK_FORMAT[MSG_TYPE_LIST['WATCHER_EVENT_MSG']],
msgData[:l])
        startTime = datetime.datetime.utcfromtimestamp(startTime)
        statusDesc = STATUS_TO_STR.get(status&STATUS_MASK, "Status Code
0x%x"%status)
        if status&IS_WATCHING_STATUS:
          statusDesc += " (watching)"
```

En este punto la conexión con el servidor ha sido correcta y se han recibido los datos. Se asignan las variables para la gestión de los datos y se comienza a trabajar con ellos. Primero se comprueba el género de la persona y posteriormente el número total de la audiencia y el género mayoritario.

```
        # Tomamos los valores de las variables
        iDWatcher=str(watcherID)
        fecha=str(startTime.year)+"-"+str(startTime.month)+"-"+str(startTime.day)
        hora=str(startTime.hour)+":"+str(startTime.minute)+":"+str(startTime.second)
        tiempo=str(duration/10)
        tiempoAtencion=str(attentionTime)
        strGenero=str(GENDER_TO_STR.get(gender, '????'))
        edad=str(AGE_TO_STR.get(age, '????'))
        strEdad=(AGE_TO_STR.get(age, '??'))

        # Anade o elimina la ID de un hombre
        if gender==1:
          #Si hay un hombre buscamos su ID en el vector mascID
          if(status==34 or status==2):
            posicion=0
```

```
    encontrado='No'
    posicion_del_valor_a_buscar=0
    longitud_del_vector=len(mascID)
    while posicion < longitud_del_vector:
      if mascID[posicion]==watcherID :
        posicion_del_valor_a_buscar=posicion
        encontrado='Si'
      posicion=posicion+1
    #Si la encontramos no hacemos nada
    if encontrado=='Si':
      pass
    #Si no, insertamos la nueva ID
    else:
      mascID.insert(posicion+1,watcherID)
  #Si el mensaje es de Watcher dead eliminamos esa ID
  else:
    mascID.remove(watcherID)

#Inserta o elimina la ID de una mujer
if gender==2:
  #Si hay una mujer buscamos su ID en el vector femID
  if(status==34 or status==2):
    posicion=0
    encontrado='No'
    posicion_del_valor_a_buscar=0
    longitud_del_vector=len(femID)
    while posicion < longitud_del_vector:
      if femID[posicion]==watcherID :
        posicion_del_valor_a_buscar=posicion
        encontrado='Si'
      posicion=posicion+1
    #Si la encontramos no hacemos nada
    if encontrado=='Si':
      pass
    #Si no, insertamos la nueva ID
    else:
      femID.insert(posicion+1,watcherID)
  #Si el mensaje es de Watcher dead eliminamos esa ID
  else:
    femID.remove(watcherID)

#Calculas la longitud de los vectores para saber numero de personas
numHombres=len(mascID)
numMujeres=len(femID)
total=numHombres+numMujeres
#Pasamos los datos a string y luego lo comparte con Scala
strTotal=str(total)
hombres=str(numHombres)
```

```
        mujeres=str(numMujeres)
        scalavar.channel_hombres=hombres
        scalavar.channel_mujeres=mujeres

        #Si hay el mismo numero de hombres y mujeres se comparte
        #el genero 0 y como mayoria "No hay mayoria"
        if numHombres==numMujeres:
          scalavar.channel_genero=0
          scalavar.channel_variable="No hay mayoria"

        #Si hay mas hombres se comparte el 1 y genero mayoritario "Masculino"
        elif numHombres>numMujeres:
          scalavar.channel_genero=1
          scalavar.channel_variable="Masculino"

        #Si hay mas hombres se comparte el 2 y genero mayoritario "Femenino"
        elif numHombres<numMujeres:
          scalavar.channel_genero=2
          scalavar.channel_variable="Femenino"

        #Si no hay audiencia comparte con Scala "No hay audiencia"
        if total==0:
            scalavar.channel_strCuenta="No hay audiencia"
            scalavar.channel_age="No hay audiencia"

        #Si hay una sola persona se comparte "1 persona" en audiencia y la
        #edad de esta persona
        elif total==1:
            scalavar.channel_strCuenta="1 persona"
            scalavar.channel_age=edad

        #Si hay mas de 1 persona comparte el numero de personas
        else:
            scalavar.channel_strCuenta=strTotal+" personas"
            scalavar.channel_age="Hay "+strTotal+" personas"

        #Tomamos el nombre del contenido que se esta mostrando
        content=scalavar.channel_contenido

        #Formamos la linea que se escribira en el archivo log.txt
        linea=(str(watcherID)+","+fecha+","+hora+","+tiempo+","+tiempoAtencion \
              +","+strGenero+","+edad+","+str(content)+","+hombres+","+mujeres+"\n")

        #Se ejecuta la rutina que escribe en el archivo
        grabartxt()
        #Esperamos 0.5 segundos para hacer la siguiente comproba
        time.sleep(0.5)
    else:
```

```
            sys.stdout.write("[%s] WATCHER_EVENT_MSG received: payload too small\n" %
ts)
except Exception, e:
   sys.stderr.write("Caught Exception: '%s', quitting .." % str(e))
```

6.3 INTEL AIM SUITE

En el caso de Intel AIM, solo proporciona el SDK en lenguaje Java por lo que se ha tenido que desarrollar un archivo Python que tomase los datos proporcionados por el archivo Java, los gestionara y se comunicara con Scala. Es decir, se ha creado un archivo java que se encargará de la conexión con el servidor y la recepción de los datos, y un archivo Python para la gestión de éstos y la posterior comunicación con la herramienta Scala.

Al igual que en el desarrollo del módulo con la herramienta Quividi, se ha creado un archivo Python que se encargará de crear y escribir en un archivo de texto llamado *log.txt.*

```
#!/usr/bin/env python
#
# Importamos los módulos que usaremos
#
import sys
import time
import os.path

#Comprueba si existe la carpeta
if not os.path.isdir('W:\AudienceLogs'):
   #Si no existe la crea, crea el archivo y escribe el encabezado
   os.mkdir('W:\AudienceLogs')
   archi=open('W:\AudienceLogs\log.txt','a',0)
   time.sleep(1)
   archi.write("IDWatcher, Fecha, Hora, Tiempo, Tiempo Atencion, Genero, Rango de Edad,
Contenido, Numero de Hombres, Numero de Mujeres\n")
else:
   #Si existe la carpeta, crea o abre el archivo y escribe el encabezado
   archi=open('W:\AudienceLogs\log.txt','a',0)
   archi.write("IDWatcher, Fecha, Hora, Tiempo, Tiempo Atencion, Genero, Rango de Edad,
Contenido, Numero de Hombres, Numero de Mujeres\n")
```

La herramienta Intel AIM trabaja de manera distinta a Quividi, mientras Quividi envía un mensaje con los datos de cada una de las personas de la audiencia, Intel AIM envía un solo mensaje con todos los datos. Es decir, si la audiencia fuese de 10 personas, Quividi enviaría 10 mensajes mientras que Intel AIM enviaría un solo mensaje con todos los datos de la audiencia. Esto implica que no sabremos la longitud del mensaje enviado por el servidor ya que depende del número de la audiencia. Para solventar esto, se realizan dos peticiones al servidor. La primera nos devuelve en número total de la audiencia, el mensaje recibido tiene una longitud fija independientemente del número total de la audiencia. La segunda petición será los datos de dicha audiencia, como ya tenemos de antemano el número total de la audiencia sabremos exactamente el mensaje que vamos a recibir.

La configuración del mensaje recibido con los datos de la audiencia es la siguiente:

Campo	Longitud	Descripción
ID	4 bytes	Valor entero que define a cada individuo
Gender	1 byte	Define el género
Age	1 byte	Define la edad
Reserved	1 byte	Reservado para uso futuro
Reserved 2	1 byte	Reservado para uso futuro
ViewingTime	4 bytes	Define el número de milisegundos que la persona ha estado delante de la cámara
Top-left X	2 bytes	Posición de la cara en el eje X
Top-left Y	2 bytes	Posición de la cara en el eje Y
Face Width	2 bytes	Ancho de la cara
Face Height	2 bytes	Altura de la cara

Para el género, el valor 0 representa un genero desconocido, 1 representa a un hombre, y 2 representa a una mujer. Para el caso de la edad, un valor de 0 representa edad desconocida, 1 representa un niño (menor de 16), 3 representa un joven adulto (16-34), 4 representa un adulto mayor (35-64), y 5 representa un anciano (mayor de 65). El valor 2, representa un adolescente (13-19) pero este rango de edad no está disponible actualmente.

Como se puede observar por cada persona de la audiencia el servidor envía 20 bytes, como tenemos el número total de la audiencia lo único que se debe hacer es multiplicar este número por 20, así se obtiene la longitud total del mensaje a recibir. Es decir, si la primera petición nos devuelve un número total de 5 personas, el mensaje a recibir por la segunda petición será de 100 bytes.

Estas dos peticiones se llevan a cabo en el archivo Java. Al igual que anteriormente se puede dividir el código, en este caso; conexión con el servidor, recepción de datos y

conexión con archivo Python. La conexión con el archivo Python se lleva a cabo mediante un archivo de texto que sirve como intermediario. El archivo Java escribe los datos en ese archivo txt y el archivo Python los lee posteriormente.

```
//Importamos los modulos que utilizaremos
import java.io.*;
import java.net.*;
import java.util.*;

public class AIMViewConnectTest {
  //Variables del sistema
  public static final int MESSAGE_GET_AUDIENCE_STATUS = 0;
  public static final int MESSAGE_GET_AUDIENCE_DETAILS= 1;
  public static final int EVENT_ACK = 128;
  public static final int EVENT_NACK = 129;
  public static final int EVENT_AUDIENCE_STATUS = 130;
  public static final int EVENT_AUDIENCE_DETAILS = 131;
  //Variables necesarias para la conexion con python
  public static int audiencia = 0;
  public static int longitud=0;
  public static String edad="";
  public static String genero="";

  public static void main(String[] args) throws InterruptedException
  {
    try
    {
    while(true){
      // Abrimos un socket en localhost, puerto 12500
      Socket myClient;
      myClient = new Socket("localhost", 12500);
      DataInputStream input;
      DataOutputStream output;
      //Creamos Stream de entrada y otro de salida
      input = new DataInputStream(myClient.getInputStream());
      output = new DataOutputStream(myClient.getOutputStream());
      FileWriter fichero = null;
      PrintWriter pw = null;

          Primera petición de datos.

      //Primero construimos el mensaje de configuracion que enviaremos al servidor
      byte request[] = new byte[5];
      request[0] = (byte)0xFA; // Magic word (primer byte)
      request[1] = (byte)0xCE; // Magic word (segundo byte)
      request[2] = (byte)0x01; // Version
      request[3] = (byte)MESSAGE_GET_AUDIENCE_STATUS; // Comando
```

```
request[4] = (byte)0x00; // Payload
output.write(request, 0, 5); // Enviamos la peticion
boolean reading = true;
int bytesRead = 0;
//Para apiGetAudienceStatus tenemos que leer los 5 bytes de la cabecera +
//1 para el payload
byte response[] = new byte[6];
//Hacemos un bucle para leer los 6 bytes
while(reading && bytesRead < 6)
{
  try
  {
    response[bytesRead] = input.readByte();
    bytesRead++;
  }
  catch(IOException e)
  {
    reading = false;
  }
}

//Verificamos si la Magic Word es correcta
if(response[0] == (byte)0xFA && response[1] == (byte)0xCE)
{
  // Verificamos que la version sea correcta
  if(response[2] == (byte)0x01)
  {
    // Verificamos q obtenemos la respuesta deseada (EVENT_AUDIENCE_STATUS)
    if(response[3] == (byte)AIMViewConnectTest.EVENT_AUDIENCE_STATUS)
    {
      // Verificamos que el tamaño del payload es correcto
      if(response[4] == (byte)1)
      {
        //La audiencia estará en el payload de la respuesta
        //(elemento 5 del vector)
        audiencia = response[5];
        //Calculamos la longitud del vector segun la audiencia
        //para el futuro mensaje de respuesta con los detalles
        longitud= 1 + (audiencia * 20);
      }
      else
      {
        System.out.println("Incorrect payload size: " + response[4]);
      }
    }
    else
    {
      System.out.println("Incorrect response message: " + response[3]);
```

```
    }
  }
  else
  {
    System.out.println("Incorrect version: " + response[2]);
  }
}
else
{
  System.out.println("Incorrect magic word: 0x" + response[0] + response[1]);
}

//Si no hay audiencia creamos el archivo que servira de conexion
//con python y escribimos FIN
if (audiencia<1) {
    //Abrimos el archivo Java2Python.txt
    fichero = new FileWriter("W:/AudienceLogs/prueba.txt");
    pw = new PrintWriter(fichero);
    pw.println("FIN");
    System.out.println("No hay audiencia");
    pw.close();
}else{

     Segunda petición de datos

//Si hay audiencia volvemos a enviar un mensaje al servidor
//pidiendo los detalles de la audiencia
request[0] = (byte)0xFA; // Magic word (primero byte)
request[1] = (byte)0xCE; // Magic word (segundo byte)
request[2] = (byte)0x01; // Version
request[3] = (byte)MESSAGE_GET_AUDIENCE_DETAILS; // Comando
request[4] = (byte)0x00; // Payload
output.write(request, 0, 5); // Enviamos la peticion
reading = true;
bytesRead = 0;
//Creamos un vector para la respuesta teniendo en cuenta la
//longitud calculada anteriormente + 5 bytes de la cabecera
byte response2[] = new byte[longitud+5];
//Hacemos un bucle para leer todos los bytes necesarios
while(reading && bytesRead < longitud+5)
{
  try
  {
    response2[bytesRead] = input.readByte();
    bytesRead++;
  }
  catch(IOException e)
  {
```

```
        reading = false;
      }
    }
    // Verificamos si la Magic Word es correcta
    if(response2[0] == (byte)0xFA && response2[1] == (byte)0xCE)
    {
      // Verificamos que la version sea correcta
      if(response2[2] == (byte)0x01)
      {
        //Verificar que obtenemos la respuesta deseada (EVENT_AUDIENCE_DETAILS)
        if(response2[3] == (byte)AIMViewConnectTest.EVENT_AUDIENCE_DETAILS)
        {
          // Verificamos que el tamaño del payload es correcto
          if(response2[4] == (byte)longitud)
          {
            //Calculamos cuantas personas forman parte de la audiencia
            //para obetener la informacion
            int i=new Double(longitud/20).intValue();
            int j=0;
            //la ID se encuentra en el elemento 9 del vector
            int id=9;
            //el genero en el elemento 10
            int gender=10;
            //la edad en el elemento 11
            int age=11;
            //el tiempo se divide en 4 bytes y esta en los elementos 14, 15, 16 y 17
            int time1=14;
            int time2=15;
            int time3=16;
            int time4=17;
            //Obtenemos la fecha y hora del sistema
            Calendar Cal = Calendar.getInstance();
            String fec=Cal.get(Cal.DATE)+"/"+(Cal.get(Cal.MONTH)+1)+"/"+Cal.get(Cal.YEAR);
            String hora=Cal.get(Cal.HOUR_OF_DAY)+":"+Cal.get(Cal.MINUTE)+":"+
                  Cal.get(Cal.SECOND);
            //Abrimos el archivo Java2Python.txt
            fichero = new FileWriter("W:/AudienceLogs/Java2Python.txt");
            pw = new PrintWriter(fichero);
            //Creamos un bucle para obtener los datos de la audiencia
            do{
              //calculamos el tiempo
              int time = (((response2[time1] << 24) & 0xff000000) | ((response2[time2] << 16) &
                  0x00ff0000) | ((response2[time3] << 8) & 0x0000ff00) | (response2[time4] &
                  0x000000ff));
              //pasamos la ID a entero
              int intID =(response2[id] & 0x000000ff);
              //pasamos el valor del genero a string
              switch (response2[gender]){
```

```
        case 0:
            genero= "0";
            break;
        case 1:
            genero="1";
            break;
        case 2:
            genero="2";
            break;
    }
    //pasamos el valor de la edad a string
    switch (response2[age]){
        case 0:
            edad= "Desconocido";
            break;
        case 1:
            edad="Child";
            break;
        case 2:
            edad="Teenager";
            break;
        case 3:
            edad="Young Adult";
            break;
        case 4:
            edad="Adult";
            break;
        case 5:
            edad="Senior";
            break;
    }
```

Escritura de los datos en el archivo txt que sirve como intermediario.

```
    //imprime los datos en el archivo
    pw.println(intID+";"+fec+";"+hora+";"+genero+";"+edad+";"+time);
    //sumamos 20 a cada valor porque cada persona
    //reporta 20 bytes de informacion
    id=id+20;
    gender=gender+20;
    time1=time1+20;
    time2=time2+20;
    time3=time3+20;
    time4=time4+20;
    age=age+20;
  } while (j<i & id<longitud & gender<longitud & time1<longitud);
  //cuando se acabe los datos imprime FIN
  pw.println("FIN");
```

```
            //Cerramos el archivo
            fichero.close();
          }
          else
          {
            System.out.println("Incorrect payload size: " + response[4]);
          }
        }
        else
        {
          System.out.println("Incorrect response message: " + response[3]);
        }
      }
      else
      {
        System.out.println("Incorrect version: " + response[2]);
      }
    }
    else
    {
      System.out.println("Incorrect magic word: 0x" + response[0] + response[1]);
    }
    }
    //Cerramos los stream
    input.close();
    output.close();
    //Esperamos 0.5 segundos antes de realizar la siguiente comprobacion
    Thread.currentThread().sleep(500);
    }
   }
   catch(IOException exp)
   {
   }
  }
}
```

El siguiente paso es la gestión de los datos y la comunicación con Scala mediante el archivo Python. Este archivo se puede dividir en lectura del archivo txt, gestión de los datos y comunicación con Scala, como en códigos anteriores, tiene un pequeño código inicial de inicialización de variables y rutinas para la escritura en el archivo de texto *log.txt*.

```
#!/usr/bin/env python
#
# Importar modulos que utilizaremos
#
import sys
```

```
import datetime
import time
import socket
import struct
import os.path
from scala5 import *

#Esperamos 5 segundos para que la plantilla se cargue
time.sleep(5)
#Ejecutamos el archivo INTELAIM.jar
os.system('java -jar INTELAIM.jar')
#Rutina que escribe en el archivo log.txt
def grabartxt():
    archi=open('W:\AudienceLogs\log.txt','a',0)
    archi.write(linea)
    archi.close()

#Devuelve el genero correspondiente en formato string
GENDER_TO_STR = { 0: 'Desconocido',
                  1: 'Hombre',
                  2: 'Mujer',
                }

# Recuperamos las variabels compartidas de Scala
scalavar = sharedvars()

#Iniciamos las variables
linea2=""
masc=0
fem=0
iDWatcher=""
id2=""
fecha=""
hora=""
tiempo=""
tiempo2=""
strGenero=""
edad2=""
while (1):
    #Abrimos el archivo de conexion con java
    archi=open('W:\AudienceLogs\Java2Python.txt','r')
    mascID=[]
    femID=[]
    descID=[]
    posMasc=0
    posFem=0
    posDesc=0
    #Leemos el archivo
```

```
lineas=archi.readlines()
for line in lineas:
    #Comprobamos que la no sea FIN
    if (line!='FIN\n' and line!='num\n'):
        #separamos la linea en cada ;
        datos = line.split(';')
        #Guardamos cada datos en una variable representativa del dato
        iDWatcher=datos[0]
        watcherID=int(iDWatcher)
        fecha=datos[1]
        hora=datos[2]
        genero=datos[3]
        gender=int(genero)
        edad2=datos[4]
        tiempo=datos[5]
        numTiempo=float(tiempo)/1000
        tiempo=str(numTiempo)
        #Comprobamos el genero y ejecutamos para que devuelva en formato texto
        strGenero=str(GENDER_TO_STR.get(gender, '????'))
        #Comprueba el genero y lo inserta en el vector correspondiente
        if gender==1:
            mascID.insert(posMasc,watcherID)
            posMasc=posMasc+1
        elif gender==2:
            femID.insert(posFem,watcherID)
            posFem=posFem+1
        else:
            descID.insert(posDesc,watcherID)
            posDesc=posDesc+1
    else:
        print "YA"

#Calculas la longitud de los vectores para saber numero de personas
numHombres=len(mascID)
numMujeres=len(femID)
numDesconocidos=len(descID)
total=numHombres+numMujeres+numDesconocidos
#Pasamos los datos a string y luego lo comparte con Scala
strTotal=str(total)
hombres=str(numHombres)
mujeres=str(numMujeres)
scalavar.channel_hombres=hombres
scalavar.channel_mujeres=mujeres

#Si hay el mismo numero de hombres y mujeres se comparte
#el genero 0 y como mayoria "No hay mayoria"
if numHombres==numMujeres:
    scalavar.channel_genero=0
```

```
    scalavar.channel_variable="No hay

  #Si hay mas hombres se comparte el 1 y genero mayoritario "Masculino"
  elif numHombres>numMujeres:
    scalavar.channel_genero=1
    scalavar.channel_variable="Masculino"

  #Si hay mas hombres se comparte el 2 y genero mayoritario "Femenino"
  elif numHombres<numMujeres:
    scalavar.channel_genero=2
    scalavar.channel_variable="Femenino"

  #Si no hay audiencia comparte con Scala "No hay audiencia"
  if total==0:
    scalavar.channel_strCuenta="No hay audiencia"
    scalavar.channel_age="No hay audiencia"
    content=scalavar.channel_contenido

  #Si hay una sola persona se comparte "1 persona" en audiencia y la
  #edad de esta persona
  elif total==1:
    scalavar.channel_strCuenta="1 persona"
    scalavar.channel_age=edad2
    content=scalavar.channel_contenido

  #Si hay mas de 1 persona comparte el numero de personas
  else:
    scalavar.channel_strCuenta=strTotal+" personas"
    scalavar.channel_age="Hay "+strTotal+" personas"
    content=scalavar.channel_contenido

  #Formamos la linea que se escribira en el archivo log.txt
linea=(iDWatcher+","+fecha+","+hora+","+tiempo+","+strGenero+","+edad2+","+str(content)+
","+hombres+","+mujeres+"\n")

  #Si hay personas y la linea a escribir no es igual que la ultima que se
  #escribio se escribe en el texto y tomamos los nuevos valores para futuras
  #comprobaciones
  if (total!=0 and (id2!=iDWatcher or tiempo2!=tiempo)):
    grabartxt()
    id2=iDWatcher
    tiempo2=tiempo
  #Esperamos 0.5 segundos para hacer la siguiente comprobacion
  time.sleep(0.5)
```

Capítulo 7. Tutorial

Tutorial para la configuración del módulo software de interacción entre herramientas de medición de audiencias y la plataforma de gestión de contenidos Scala.

7.1 Configuración de los softwares

Para llevar a cabo el software de integración necesitamos varios softwares, vamos a ver como configurar cada uno de ellos para el buen funcionamiento.

7.1.1 Quividi

Una vez instalado el software de Quividi detectará nuestro sensor óptico de manera automática. Si tuviésemos más de uno debemos ir a Quividi Control Center (http://localhost:81/config.html), y cambiarlo en el apartado *Video Source*. En esta página podemos modificar el modo de trabajo en el apartado *Mode of Operation.* Para nuestro caso particular deberemos elegir la opción *VidiReports (viewers/attention)* (Figura 7.1). Una vez tengamos esta parte configuramos debemos ir a *Config file* dentro de *Config* y habilitar el socket. Para ello, debemos cambiar en el archivo el 0 por un 1, quedando: *enable_socket=1.* (Figura 7.2).

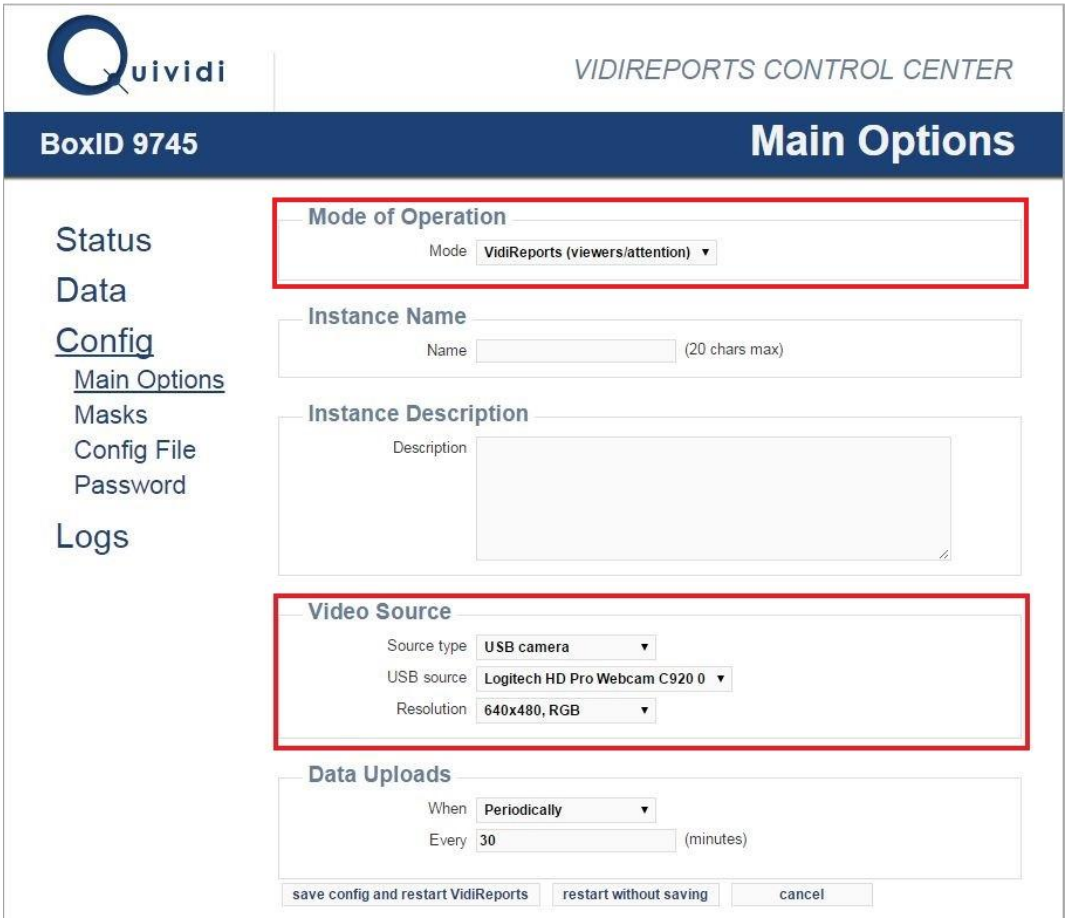

Figura 7.1. Captura de VidiReports Control Center, elaboración propia

Figura 7.2. Captura del Config File habilitanto el socket, elaboración propia.

7.1.2 Intel AIM

Intel AIM, al contrario que Quividi, no detecta nuestro sensor de forma automática. Para configurarlo, primero debemos ir al AIM Manage y copiar el License Code (Figura 7.3). Una vez y tenemos este código debemos ejecutar el software AIMSuite y en la pestaña *AIMSensor* pulsamos *Configure AIMSensor* (Figura 7.4). En la pestaña *License* introducimos el License Code y copiamos el Activation Code, ya que deberemos introducirlo en el AIM Manage para configurar completamente nuestro sensor óptico (Figura 7.5).

Figura 7.3. Captura de AIM Manage, elaboración propia.

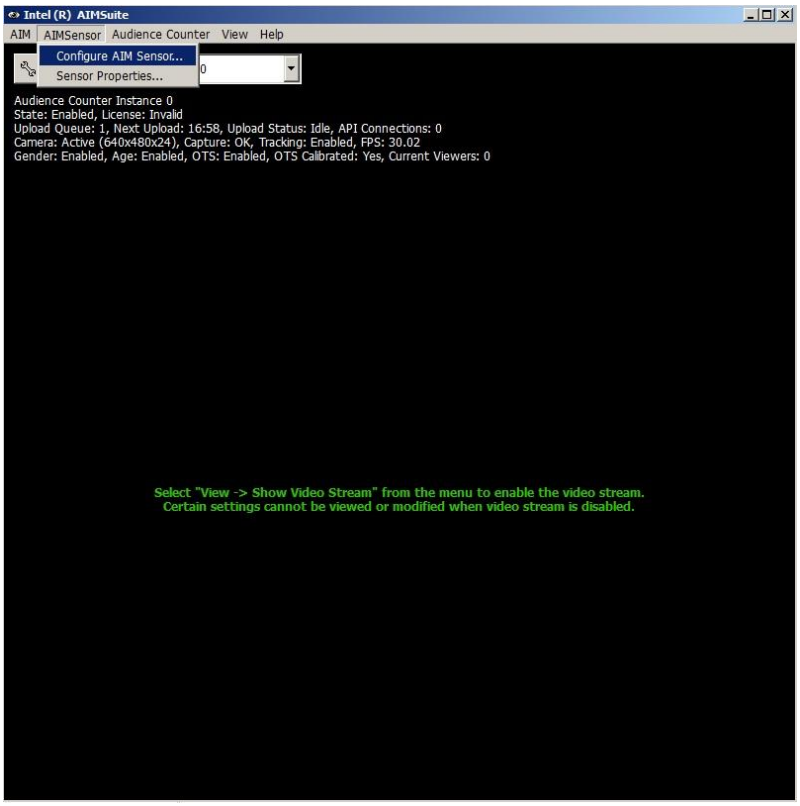

Figura 7.4. Captura de AIM Suite, elaboración propia.

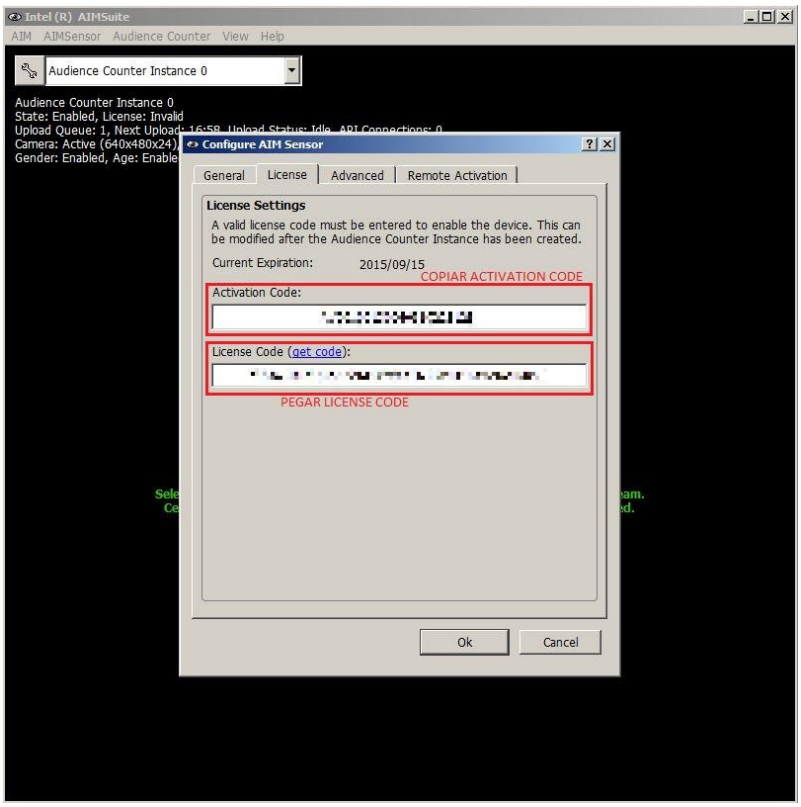

Figura 7.5. Captura de Configure AIM Sensor, elaboración propia.

7.2 Creación de guion en Scala

Una vez tenemos los softwares de medición y captación configurados, el proceso a llevar a cabo es muy similar. La diferencia reside en los archivos que utilizaremos, en el momento en el que tengamos que hacer uso de ellos se explicará que archivo.

Cabe destacar que Scala nos permite trabajar de dos maneras; a través del Content Manager y un Player, o desde un ordenador. Las pruebas finales se han realizado utilizando un ordenador, pero se han creado los guiones necesarios para que funcione a través de Content Manager y el Player. Dado que los cambios a realizar son mínimos explicaremos como crear el guion utilizado para las pruebas y, posteriormente, explicaremos los cambios a realizar para que funciones a través del Content Manager y el Player.

Primero hemos creado una plantilla con una apariencia que facilitara la compresión de los datos reproducidos en la pantalla con el Adobe Photoshop (Figura 7.6) y lo exportamos de tal manera que tengamos la plantilla dividida en capas (Figura 7.7).

Figura 7.6. Plantilla creada en Photoshop, elaboración propia.

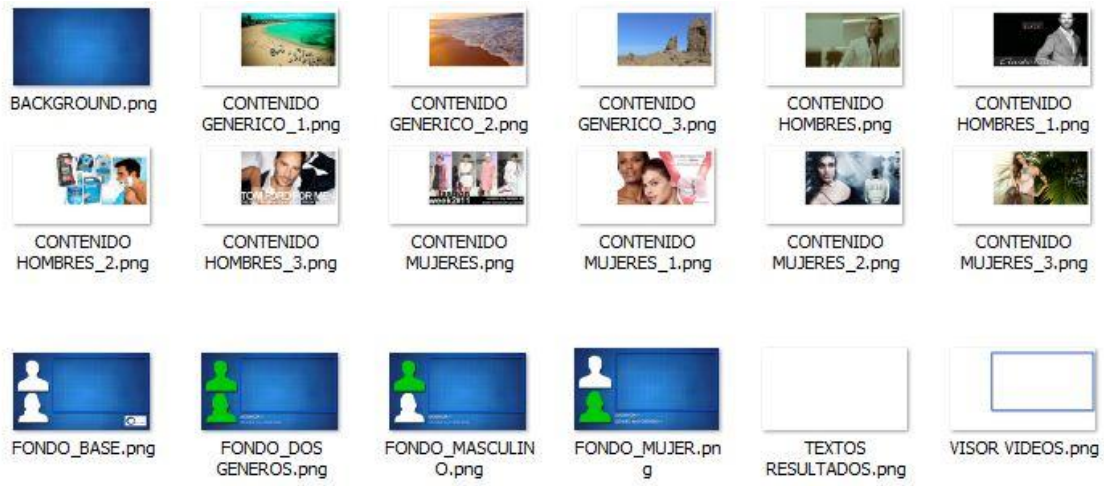

Figura 7.7. Plantilla dividida en capas, elaboración propia.

Una vez y tenemos la plantilla, vamos al Scala Designer. Pulsamos en la pestaña *Agregar* y *Agregar página normal*. Damos doble click en la nueva página y pulsamos en la pestaña *Agregar* y en *Agregar archivos...*, elegimos todos los archivos creados anteriormente y los colocamos correctamente.

Después, debemos inicializar las variables que utilizaremos, esto lo hacemos de la siguiente forma. Pulsamos la pestaña *Agregar* y pulsamos en *Agregar evento especial*. Tenemos que crear un *evento especial* por cada variable que queramos crear. En nuestro caso creamos seis eventos especiales, ya que tenemos seis variables, a saber;

- channel.genero (tipo integer) que indica el género mayoritario en la audiencia.
- channel.hombres (tipo String), indica una cadena que muestra el número de hombres de la audiencia.
- channel.mujeres (tipo String), indica el número de mujeres en la audiencia.
- channel.variable (tipo String), se trata del género mayoritario en formato texto.
- channel.age (tipo String), si la audiencia consta de una sola persona nos muestra su edad.
- channel.strCuenta (tipo String), nos muestra el número total de la audiencia.

Para iniciar las variables vamos al campo *Variable* y en la pestaña *Definir variable* (Figura 7.8) escribimos el nombre de la variable y el valor inicial según el tipo de variable. Si es tipo entero el valor inicial es 0 (channel.genero = 0) y si es tipo String es valor inicial será un valor entre comillas (channel.hombres = “0”).

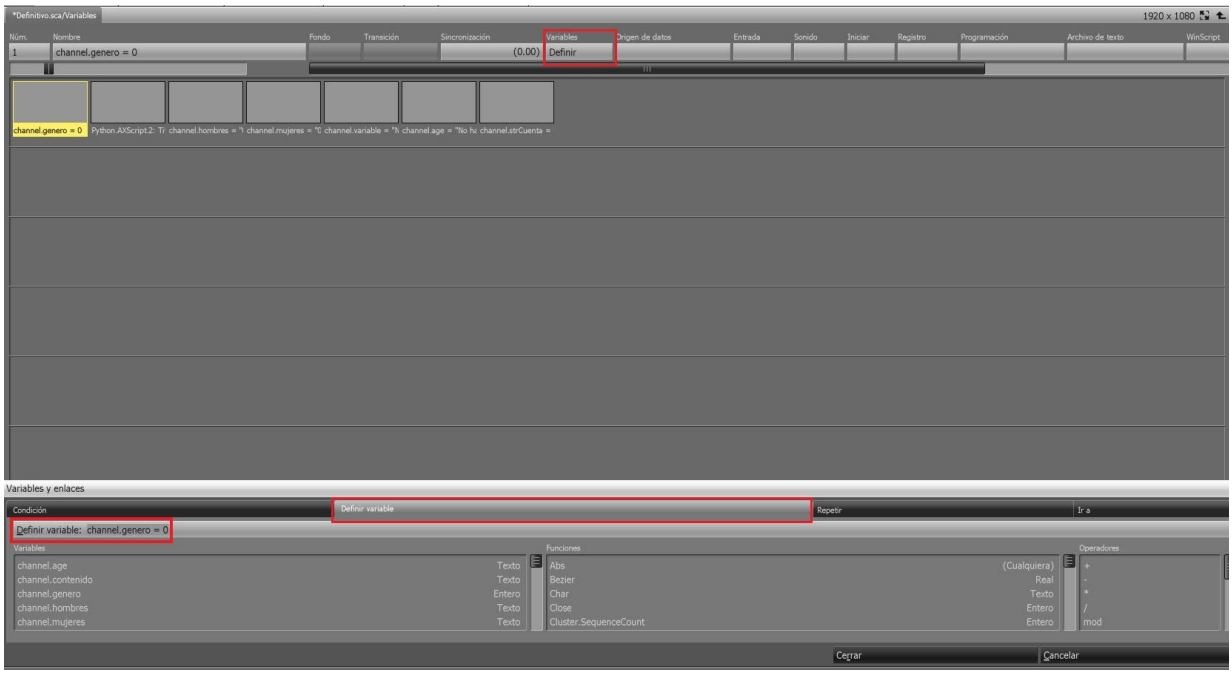

Figura 7.8. Captura de Scala Designer, elaboración propia

Para el buen funcionamiento de la plantilla debemos añadir condiciones como "*SI"* o "*Ir a"* a los elementos de ésta. Por ejemplo, supongamos que la audiencia es mayoritariamente masculina, en este caso la silueta de hombres debe marcarse en verde y el contenido a mostrar será el contenido para hombres. Para llevar a cabo eso hacemos lo siguiente, pulsamos en el icono *Lista* para que nos muestre una lista con todos los elementos de la plantilla, marcamos la silueta verde de hombres y pulsamos en el campo *Variables* (Figura 7.9). En este momento aparecen cuatro pestañas, vamos a la pestaña *Condición* y ahí debemos poner que condición será necesaria para que se muestre la silueta verde de hombre. Si pensamos un poco nos damos cuenta que la condición indispensable es que haya uno o más hombres en la audiencia por lo que pondremos como condición *channel.hombres<>"0"* (Figura 7.10), para el caso de la silueta verde de mujer la condición es *channel.mujeres<>"0"*.

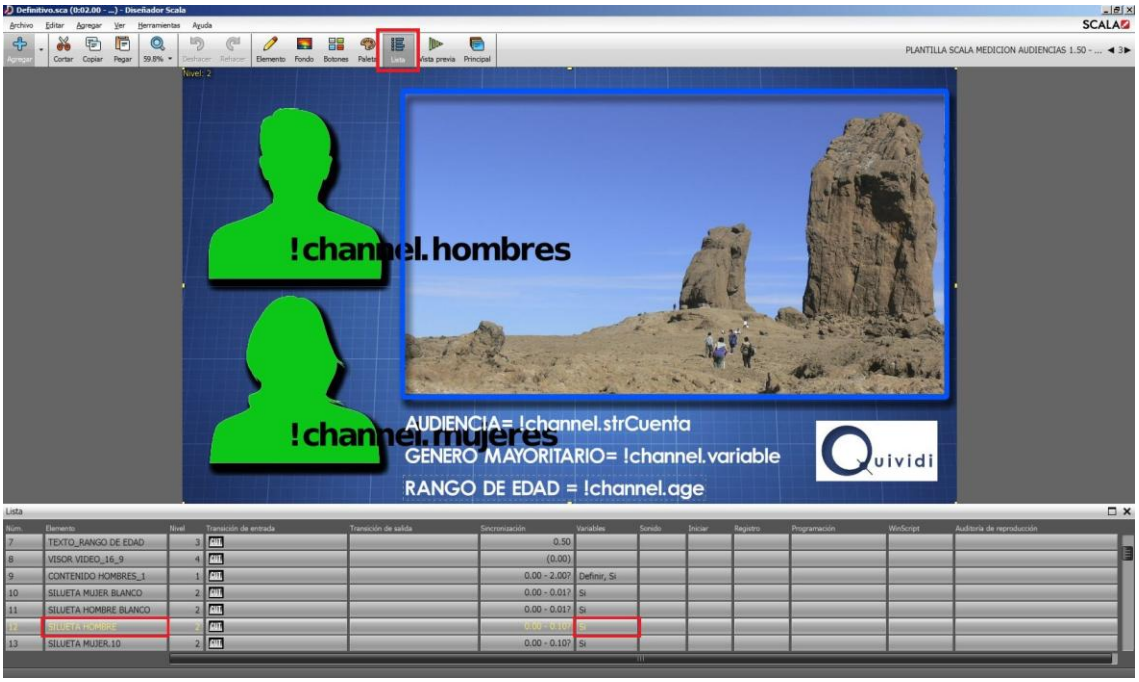

Figura 7.9. Ejemplo para crear condición, elaboración propia.

Figura 7.10. Definiendo condición, elaboración propia.

Una vez y tenemos los elementos de la plantilla colocados correctamente y las variables iniciadas, debemos realizar unos pasos para que se presenten en pantalla el valor de las variables que estamos utilizando. Para llevar a cabo esto, pulsamos en la pestaña *Agregar* y en *Agregar texto* y aparecerá el cursor en la plantilla para escribir el texto (Figura 7.11). Para que nos presente el valor de una variable hay que escribir el nombre de dicha variable precedida de un signo de exclamación. Por ejemplo, para el caso del número de

hombres en la audiencia debemos escribir *!channel.hombres*. Una vez y tenemos el texto pulsamos en él y lo movemos a la posición deseada, en este caso encima de la silueta del hombre (Figura 7.12). Hay que tener en cuenta que el valor de la variable se mostrará en el principio del texto, por esto no centramos el texto encima de la silueta.

Figura 7.11. Agregar texto

Figura 7.12. Colocar texto en la plantilla

En el caso del contenido a mostrar, debemos además definir el valor de una variable. Se trata de la variable channel.contenido, esta variable se verá modificada según el contenido presentado en pantalla y, es por esto mismo, se define en cada contenido. En este caso, además de crear una condición que en este caso será *channel.genero="1"* (para el caso de contenido para mujeres la condición será *channel.genero="2"* y para contenido genérico *channel.genero="0"*), definiremos la variable channel.contenido pulsando en la pestaña *Definir variable*, en el caso del ejemplo debemos definir la variable de la siguiente manera, *channel.contenido="CONTENIDO HOMBRES 1"* (Figura 7.13).

Figura 7.13. A la izquierda, Condición y a la derecha, Definir variable, elaboración propia.

Además, debemos crear otro evento especial para un archivo Python que se encargará de crear el archivo "log.txt" donde quedan registrados todos los datos de la audiencia. En este caso vamos a la pestaña *WinScript* y pulsamos en el campo *Secuencia de Comandos de Windows*, aparecerá una ventana emergente en la que podremos buscar el archivo llamado *Title.py* (Figura 7.14).

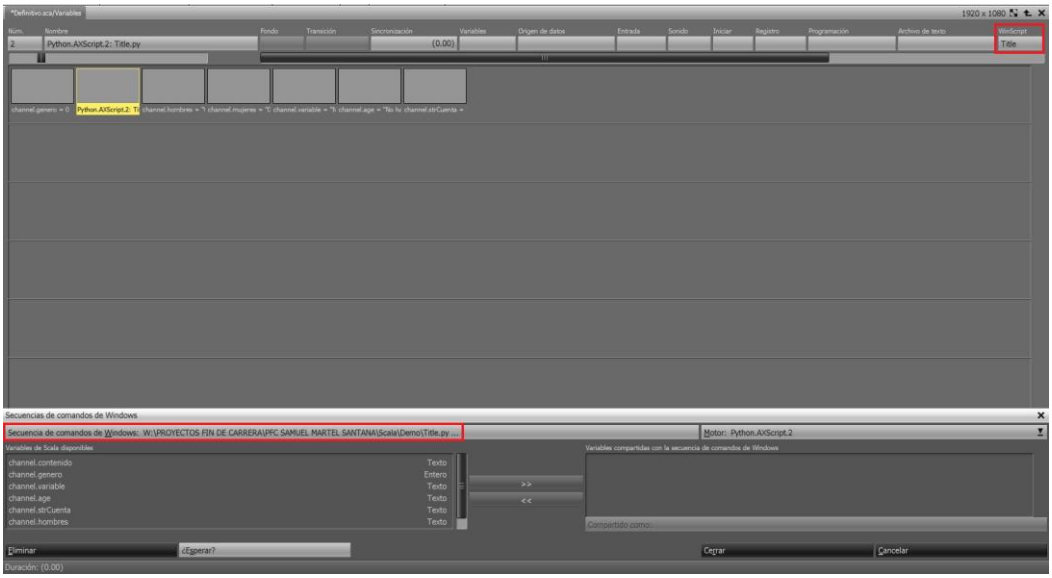

Figura 7.14. Archivo Title.py

Para organizar el guion podemos seleccionar estos siete eventos especiales y pulsando el botón secundario y pulsamos *Agrupar*, de esta manera crearemos un carpeta en la que estarán todos los eventos especiales (Figura 7.15).

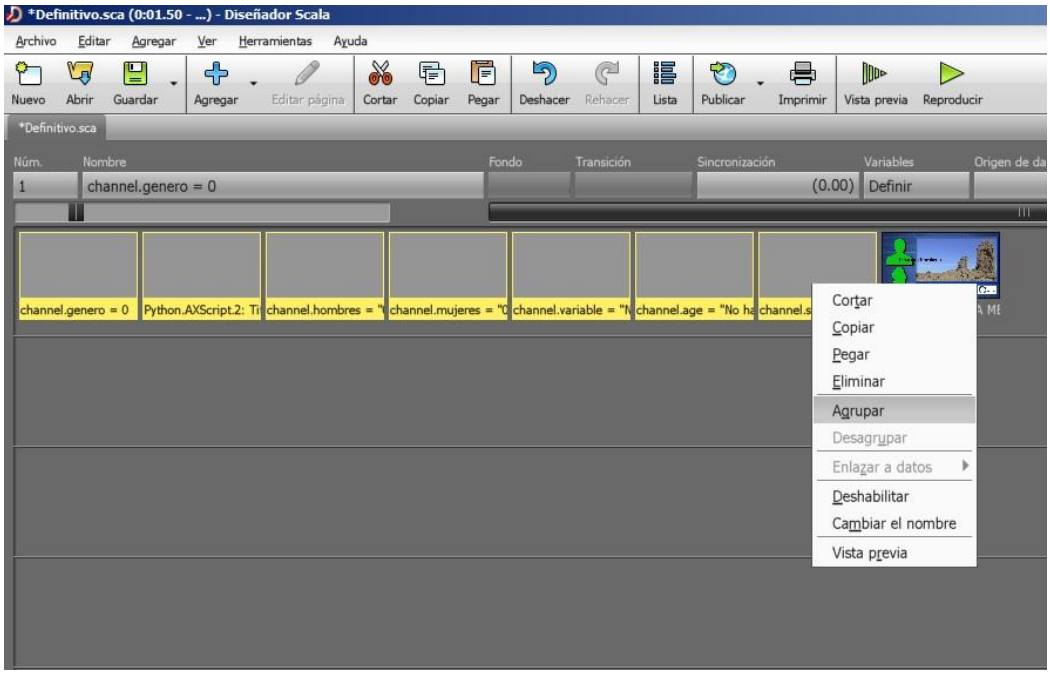

Figura 7.15. Agrupar páginas

Ya tenemos el guion casi finalizado, a estas alturas el guion debe tener la siguiente vista (Figura 7.16).

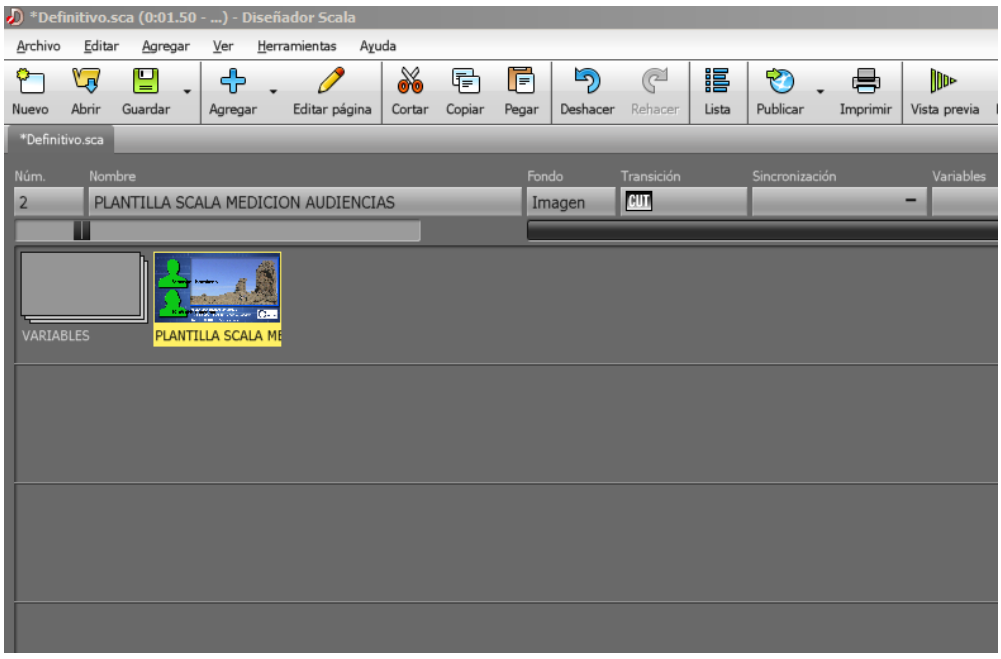

Figura 7.16. Vista del guion, elaboración propia.

Falta la parte más importante, el archivo Python. Dicho archivo se encargará de tomar los datos de la audiencia desde un servidor, procesarlos y presentar los datos por pantalla.

Como se comentó anteriormente, tenemos dos softwares y por tanto dos archivos Python distintos. Para que Scala ejecute el archivo haremos lo siguiente, creamos una página normal al igual que se hizo para la plantilla. En la pestaña *WinScript* pulsamos en el campo *Secuencia de Comandos de Windows* y buscamos el archivo llamado *Quividi.py* para el caso de usar Quividi e *IntelAIM.py* para el caso de Intel AIM (Figura 7.17).

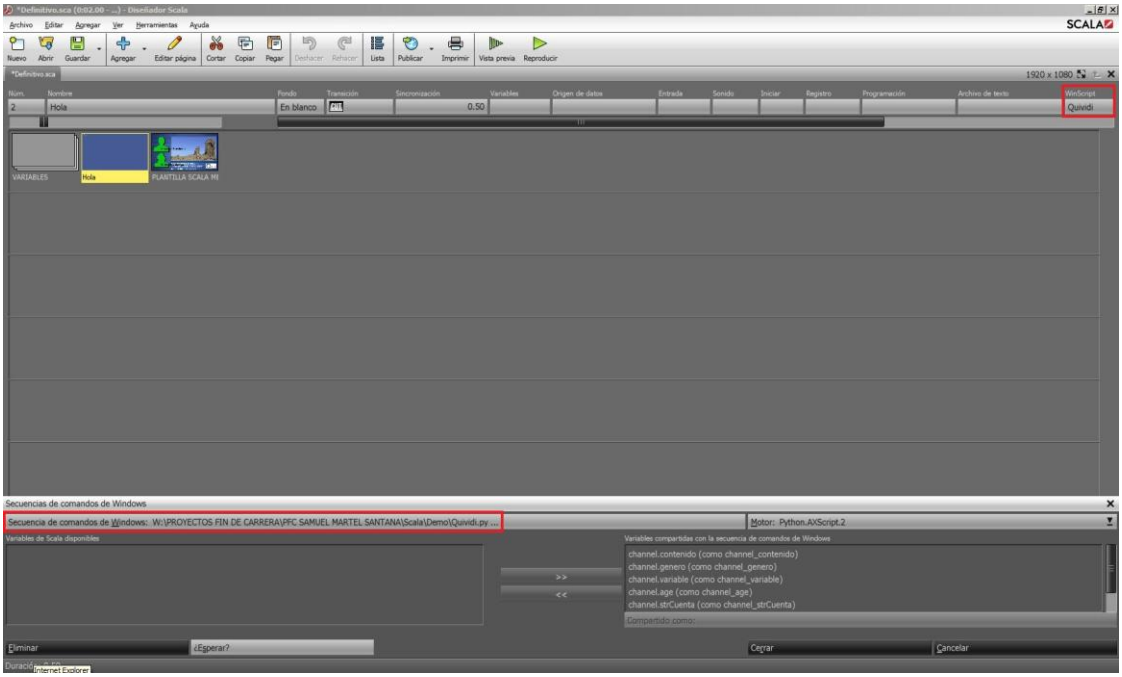

Figura 7.17. Archivo Quividi.py

Si queremos trabajar a través del Content Manager y el Player debemos tener en cuenta que debemos tener dos guiones de Scala. Esto se debe a que en el Player tendremos solo el guion que presenta los datos en pantalla y en nuestro ordenador o servidor debemos ejecutar el guion que toma los datos de la audiencia del servidor. Hay que tener en cuenta que los sensores ópticos deben tener conexión, ya sea cableada o inalámbrica, con nuestro ordenador, ya que los softwares de medición y detección estarán instalados ahí. Para que nos funcione a través del Content Manager y el Player debemos realizar los siguientes cambios.

Vamos a diferenciar los guiones con los nombres *CREAR LA VARIABLE*, que serán el guion que tendremos en nuestro ordenador y *PRESENTAR LA VARIABLE*, que será el guion que reproducimos en el Player.

En el guion *CREAR LA VARIABLE* tendremos que iniciar las variables como se comentó anteriormente y una página normal donde pondremos el archivo Python, *Quividi.py*

para el uso de Quividi e *IntelAIM.py* para el caso de Intel AIM (Figura 7.18). Además, debemos iniciar una nueva variable, *channel.contenido*. Si recordamos, esta variable se iniciaba en la plantilla pero al no tenerla en este guion es necesario iniciarla al igual que el resto de variables.

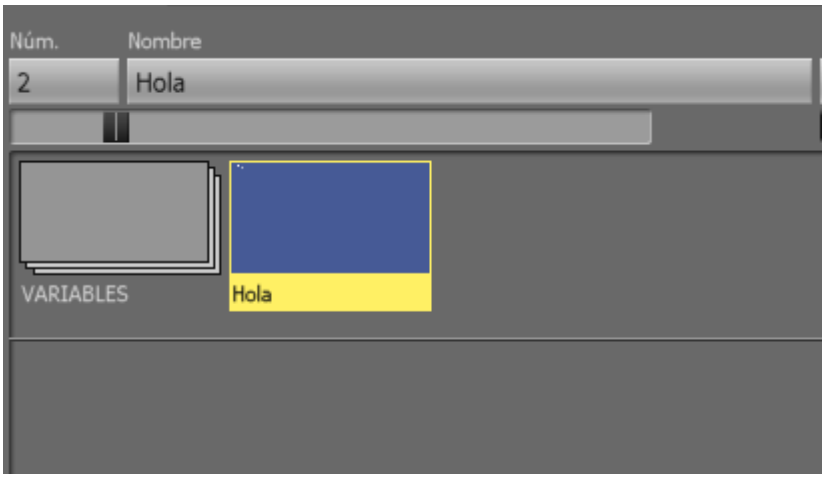

Figura 7.18. Guion CREAR LA VARIABLE

El guion *PRESENTAR LA VARIABLE* constará de los eventos especiales para iniciar las variables y la plantilla donde mostraremos los datos (Figura 7.19). El hecho de iniciar las variables en ambos guiones es porque ambos las van a usar, si solo iniciáramos las variables en uno de los guiones el otro presentaría errores.

Figura 7.19. Guion PRESENTAR LA VARIABLE

Para tener una correcta conexión entre guiones debemos crear variables de canal, es decir variables del Content Manager. En el Content Manager creamos las variables para que los guiones puedan transmitirse información entre ellos. Las variables a crear se

llamaran como las variables creadas anteriormente sin el *channel.*, es decir *channel.hombres* se creará en el Content Manager como *hombres.* A la hora acceder a ellas desde los guiones de Scala se debe complementar el nombre de la variable con *channel.*, es por esto que las variables se han creado así desde un principio. De esta manera, se facilita la creación y modificación de futuros guiones.

Capítulo 8. Pruebas y conclusiones

Para la realización de este Trabajo Fin de Grado se ha llevado a cabo una prueba para comprobar el buen funcionamiento del software de integración.

8.1 Pruebas realizadas

Para comprobar el buen funcionamiento del software de integración se han llevado a cabo pruebas en las que se situaban delante del sensor óptico distintos grupos de audiencia para, de esta manera, comprobar en la pantalla que el contenido que se muestra esté acorde con la audiencia. El esquema de la instalación para las pruebas es la siguiente.

Figura 8.1. Instalación para la realización de pruebas

Se ha colocado un televisor NEC V422 como pantalla, como sensor óptico instalamos una cámara USB Logitech C920. El gestor de contenido, así como los softwares necesarios para el buen funcionamiento de los sensores ópticos y la medición y captación de audiencia lo hemos instalado en una estación de trabajo Intel Dual Core con 4 GB RAM, HDD SATA2, Monitor 22", Win7 (Estación SERVIDOR). En las imágenes se observan dos sensores ópticos, el segundo se ha utilizado para la grabación de la audiencia para el video de demostración (Figura 7.2).

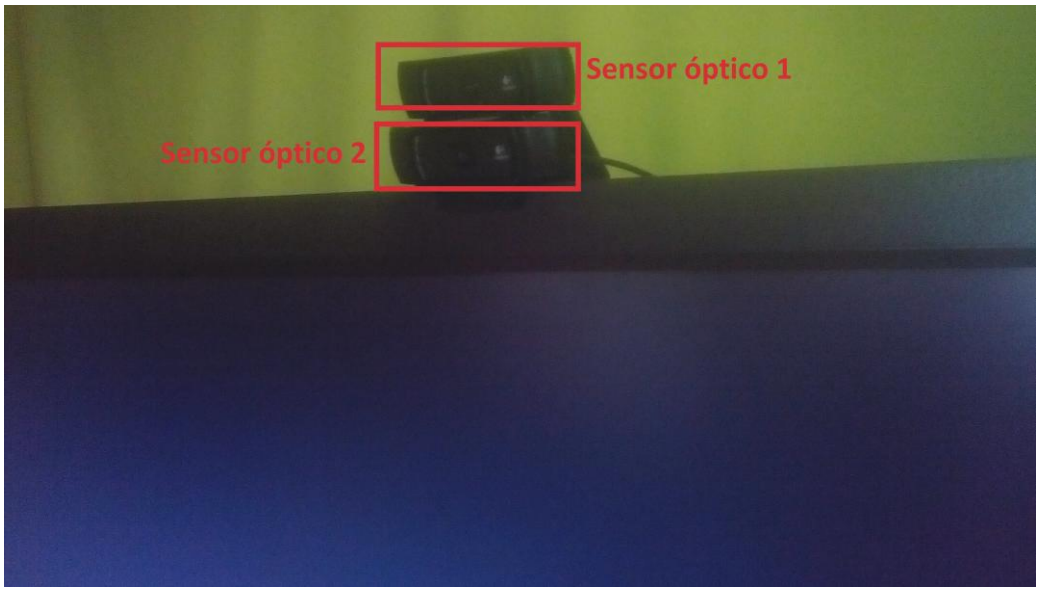

Figura 8.2. Captura de los sensores ópticos, elaboración propia.

Los elementos necesarios para la realización de dichas pruebas, tanto hardware como software, han sido los siguientes:

- Televisor NEC V422
- 2 cámaras USB Logitech C920
- Estación de trabajo basada en Intel Pentium Quad Core, 4 GB RAM utilizado como PC de diseño.
- Estación de trabajo basada en Intel Pentium Dual Core, 4 GB RAM utilizado como PC Player.
- Scala Designer
- Quividi
- Intel AIM Suite
- Mezclador Tricaster 450 Extreme.

Figura 8.3. Captura de pruebas realizadas I

Hemos ejecutado el guion de Scala desde el Designer y han ido entrando en la sala distintos grupos de audiencia intentando abarcar el mayor número de combinaciones posibles, de esta manera ha sido demostrado que el software de integración entre plataformas funciona correctamente, ya fuese una sola persona o un grupo. Las pruebas se extendieron a lo largo de 6 días, entre los que se incluyen la grabación de una demo.

Figura 8.4. Captura de pruebas realizadas II

Figura 8.5. Captura de pruebas realizadas III

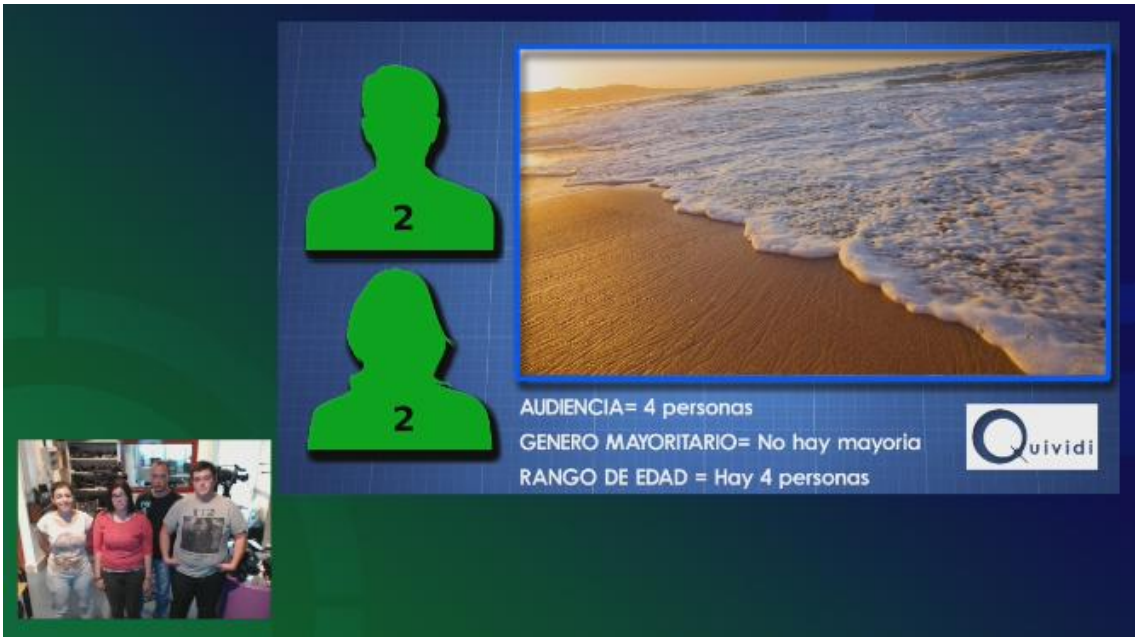

Figura 8.6. Captura de pruebas realizadas IV

Muestra de medidas obtenidas tras la realización de las pruebas en el archivo *log.txt.*

IDWatcher	Fecha	Hora	Tiempo	Genero	Rango de Edad	Contenido	Número de Hombres	Número de Mujeres
376	25/07/2014	16:41:11	2	Hombre	Joven Adulto	CONTENIDO GENERICO 2	1	0
376	25/07/2014	16:41:11	4	Hombre	Joven Adulto	CONTENIDO HOMBRES 1	1	0
376	25/07/2014	16:41:11	5	Hombre	Joven Adulto	CONTENIDO HOMBRES 2	1	0
376	25/07/2014	16:41:11	7	Hombre	Joven Adulto	CONTENIDO HOMBRES 2	1	0
376	25/07/2014	16:41:11	6	Hombre	Joven Adulto	CONTENIDO HOMBRES 3	0	0
377	25/07/2014	16:41:20	1	Hombre	Joven Adulto	CONTENIDO HOMBRES 3	1	0
377	25/07/2014	16:41:20	2	Mujer	Adulto	CONTENIDO HOMBRES 1	0	1
377	25/07/2014	16:41:20	7	Mujer	Adulto	CONTENIDO MUJERES 2	0	1
377	25/07/2014	16:41:20	8	Mujer	Adulto	CONTENIDO MUJERES 2	0	1
377	25/07/2014	16:41:20	10	Hombre	Joven Adulto	CONTENIDO MUJERES 3	1	0
377	25/07/2014	16:41:20	13	Mujer	Adulto	CONTENIDO HOMBRES 1	0	1
377	25/07/2014	16:41:20	14	Mujer	Joven Adulto	CONTENIDO HOMBRES 1	0	1
377	25/07/2014	16:41:20	16	Mujer	Joven Adulto	CONTENIDO MUJERES 1	0	1
377	25/07/2014	16:41:20	16	Mujer	Joven Adulto	CONTENIDO MUJERES 2	0	0
379	25/07/2014	16:41:48	2	Mujer	Joven Adulto	CONTENIDO GENERICO 2	0	1
379	25/07/2014	16:41:48	4	Mujer	Joven Adulto	CONTENIDO MUJERES 1	0	1
379	25/07/2014	16:41:48	5	Mujer	Joven Adulto	CONTENIDO MUJERES 1	0	1
379	25/07/2014	16:41:48	6	Mujer	Joven Adulto	CONTENIDO MUJERES 2	0	0
383	25/07/2014	16:41:59	1	Hombre	Joven Adulto	CONTENIDO GENERICO 2	1	0
383	25/07/2014	16:41:59	2	Hombre	Joven Adulto	CONTENIDO HOMBRES 1	1	0
383	25/07/2014	16:41:59	4	Hombre	Joven Adulto	CONTENIDO HOMBRES 1	1	0
383	25/07/2014	16:41:59	5	Hombre	Joven Adulto	CONTENIDO HOMBRES 2	1	0
383	25/07/2014	16:41:59	4	Hombre	Joven Adulto	CONTENIDO HOMBRES 2	0	0

8.2 Conclusiones

Tras llevar a cabo las pruebas pertinentes, hemos podido desarrollar un software de integración entre las plataformas de gestión de contenidos Scala y las herramientas de medición y detección de audiencia, Quividi e Intel AIM.

Una vez obtenidos los datos del software pudimos establecer la diferencia entre las herramientas de medición y detección de audiencia. Una vez solucionados algunos problemas iniciales a la hora de interactuar con Quividi, que han sido solventados con la nueva actualización del software, observamos como el porcentaje de error en la detección de audiencias en notablemente inferior que respecto a Intel AIM.

De inicio Quividi nos facilita el SDK para la interacción con el sensor óptico en diversos lenguajes de programación, Java, Python,… al contrario que Intel AIM que solo nos proporciona el SDK en Java, hecho que dificultó la interacción con Scala Designer ya que éste no dispone de un complemento para la ejecución de archivos Java, si así para Python. Esto nos llevó a tener que programar un archivo Python que sirviera de unión entre el archivo de interacción Java con el Scala Designer y por tanto dificultó el software de interacción.

Tanto por las herramientas facilitadas por las empresas como por el menor porcentaje de error a la hora de detectar a la audiencia, considero que Quividi es una opción más acertada como herramienta de medición y detección de audiencias en una red de DS.

Bibliografía

En este apartado se expone la bibliografía consultada para la realización de este Trabajo de Fin de Grado

[1] Informe sobre Digital Signage
http://www.itu.int/en/ITU-T/techwatch/Pages/digital-signage-standards.aspx
[2] Web del sistema de detección y medición Intel AIM https://aimsuite.intel.com/
[3] Web del sistema de detección y medición Quividi http://www.quividi.com/
[4] Web de la plataforma Scala Infochannel http://scala.com/
[5] Scala: "Best of Breed" Software. The best in the business.
http://www.hammonddigitalsignage.com/Scala/
[6] Presentation de Adrian J Cotterill, editor de DailyDOOH.com en la ISE 2009 Digital Out of Home Business Conference en Amsterdam (Fecha de la última visita 10/10/14)
https://es.scribd.com/doc/11555496/ISE-DOOH-Business-Conference-Feb-2nd-2009
[7] InPark Magazine http://www.inparkmagazine.com/smart-monkeys-designs-media-system-for-new-lax-international-terminal/
[8] Society for Experiential Graphic Design https://segd.org/
[9] *Wall Street Beat: Time to Put Off Buying LCD TVs and Displays* By DAN NYSTEDT, IDG News Service\Taipei Bureau, IDG Published: August 8, 2008
http://www.nytimes.com/idg/IDG_852573C400693880048257498001FBEC6.html?partner=rssnyt&emc=rss
[10] Baluns http://ludens.cl/Radiacti/topicos/balun/balun.htm
[11] Focus Media Holding Ltd. http://www.focusmedia.cn/en/aboutus/companyoverview.htm
[12] (2010) *NIELSEN "FOURTH SCREEN NETWORK AUDIENCE REPORT"*
http://www.nielsen.com/content/dam/corporate/us/en/newswire/uploads/2010/04/On-Location-Fourth-Screen-Report-4Q-FINAL-PR.pdf
[13] Blanch, M. (1998). *Como se miden las audiencias en radio.* Barcelona: CIMS
[14] Lamas, C. *La medición de audiencias en Europa.* AIMC
http://www.aimc.es/Lamas-Carlos-La-medicion-de.html
[15] García-Úbeda, M. (2011). *Las Claves de la Publicidad.* Madrid: ESIC EDITORIAL.
[16] García Ferrer, G. (2012). *Investigación Comercial.* Madrid: ESIC EDITORIAL.
[17] Lamas, C. *La medición de la audiencia de los medios: una visión actualizada.*
http://www.aimc.es/Lamas-Carlos-La-medicion-de-la.html
[18] *Radio-powered poster research 'economical'.* (2002)
http://www.mrnews.com/sample25.html

[19] Intel AIM Suite https://aimsuite.intel.com/what-aim-suite

[20] QUIVIDI http://www.quividi.com/vidireports.html

[21] CUENDE Infometrics http://www.cuende.com/index.htm

[22] Crystal Displays - Home http://www.crystaldisplay.com/cms/

[23] TruMedia http://www.tru-media.com/

[24] relEYEble https://portal.releyeble.com/#!/

Presupuesto

Don Samuel Ángel Martel Santana, autor del presente Trabajo de Fin de Grado, declara que:

El Trabajo de Fin de grado con título “Desarrollo de módulos software para la interacción entre herramientas de medición de audiencias y plataformas de gestión de contenidos basadas en Digital Signage”, desarrollado en la Escuela de Ingeniería de Telecomunicación y Electrónica de la Universidad de las Palmas de Gran Canaria, tiene un coste de desarrollo total de **28.469,79 €** correspondiente a la suma de las cantidades consignadas a los apartados considerados a continuación.

Las Palmas de Gran Canaria a 9 de diciembre de 2014

Firma: Samuel Ángel Martel Santana

1. Desglose del presupuesto

Para la realización del presupuesto se han seguido las recomendaciones del Colegio oficial de Ingenieros de Telecomunicación (COIT) sobre los baremos orientativos mínimos para trabajos profesionales en 2009.

Hasta el año 2008, el COIT ha venido publicando unas listas de Honorarios Orientativos que en los dos últimos años se denominaron Costes Estimados de Trabajos Profesionales. Por modificación de la Ley de Colegios Profesionales, mediante Ley 25/2009 de 22 de diciembre, no es posible seguir publicando estas listas. El Colegio no puede elaborar baremos de honorarios, ni siquiera orientativos, salvo que sean con la finalidad de tasar costes en los procedimientos judiciales. No obstante y con absoluto respeto a lo establecido en la mencionada Ley, el COIT dispone de una herramienta de cálculo para que el colegiado pueda valorar por sí mismo los trabajos profesionales que realiza, en función de la organización de su actividad. Esta herramienta se encuentra en la web del COIT en Ejercicio Profesional / Apoyo y Desarrollo Técnico / Información General.

El presupuesto se ha desglosado en varias secciones en las que se han separado los distintos costes asociados al desarrollo del TFG, fijándose la duración del mismo en 4 meses. Estos costes se dividen en:

1. Recursos materiales.
2. Trabajo tarifado por tiempo empleado.
3. Costes de redacción del Trabajo Fin de Grado.
4. Material fungible.
5. Derechos de visado del COIT.
6. Costes de tramitación y envío.
7. Aplicación de impuestos.

2. Recursos materiales

Para la ejecución de este Trabajo Fin de Grado han sido necesarias las herramientas software para la realización de medición y detección de audiencia, los softwares para la captura y el retoque fotográfico y la edición de video, así como el paquete office para la redacción de la memoria. En cuanto al hardware que se ha necesitado, un televisor, dos cámaras USB, un ordenador y una impresora. La amortización se calcula sobre el tiempo útil del recurso material. El sistema de amortización se toma como lineal, asumiendo que el

inmovilizado material se desprecia de forma constante a lo largo de su vida útil. La cuota de amortización anual se calcula usando la siguiente fórmula:

$$Cuota = \frac{Valor\ de\ adquisición - Valor\ residual}{Tiempo\ de\ vida\ útil}$$

2.1 Recursos hardware

Para la ejecución de este estudio las herramientas hardware que se han utilizado son las siguientes:

- Dos cámaras USB Logitech C920
- Estación de trabajo Intel Pentium Quad Core, 4 GB RAM utilizado como PC de diseño.
- Estación de trabajo Intel Pentium Dual Core, 4 GB RAM utilizado como PC Player.
- Televisor NEC V422
- Mezclador Tricaster 450 Extreme.
- Ordenador personal.
- Impresora.

Recurso	Valor de adquisición (€)	Valor residual (€)	Vida útil (años)	Cuota anual (€)	Uso (meses)	Cuota aplicable (€)
Dos cámaras USB	105x2	40x2	5	13x2	8	17,3
PC de diseño	1.250	600	5	130	8	86,7
PC Player	950	400	5	110	8	73,3
Televisor NEC V422	1.185	550	5	127	8	84,7
Mezclador Tricaster	19.795,50	10.000	12	816,29	8	544,19
Ordenador personal	702	300	4	100,5	8	67
Impresora	90	30	3	20	8	13,3
		TOTAL				**886,49**

Tabla P1. Recursos hardware

2.2 Recursos software

Las herramientas software necesarias son:

- Licencia Quividi (VidiReports + VidiCenter License)
- Licencia Intel AIM (AIM Suite Audience Counter Gold)
- Licencia Scala Designer (SW-IDE Scala Designer)
- Licencia Scala PC Player (SW-PLAD. Scala PC Player License)
- Adobe Photoshop CS6
- Adobe Premiere CS6
- Microsoft Office 2013

Recurso	Valor de adquisición (€)	Valor residual (€)	Vida útil (años)	Cuota anual (€)	Uso (meses)	Cuota aplicable (€)
Licencia Quividi	390	0	1	390	8	260
Licencia Intel AIM	200	0	1	200	8	133,33
Licencia Scala Designer	720	0	1	720	8	480
Licencia Scala PC Player	620	0	1	620	8	413,33
Adobe Photoshop CS6	942,18	300	5	128,43	8	85,62
Adobe Premiere CS6	1.060,26	400	5	132,05	8	88,03
Microsoft Office 2013	152,99	75	3	25.99	8	17,33
		TOTAL				**1.477,64**

Tabla P2. Recursos Software.

3 Trabajo tarifado por tiempo empleado

En este Trabajo Fin de Grado se han empleado 300 horas en las tareas de formación, especificación, desarrollo y documentación necesarias para la elaboración del mismo. El importe de las horas de trabajo empleadas para la realización del proyecto se calcula siguiendo las recomendaciones del COIT:

$$H = C_t * 74{,}88 * H_n + C_t * 96{,}72 * H_e$$

Donde:

- H son los honorarios totales por el tiempo empleado.
- H_n son las horas normales trabajadas (dentro de la jornada laboral).
- H_e son las horas especiales.
- C_t es un factor de corrección función del número de horas trabajadas.

Para la realización de este Trabajo Fin de Grado se han necesitado 300 horas (3h/día*100 días), todas ellas dentro del horario normal.

Según el COIT, el coeficiente C_t tiene un valor variable en función del número de horas empleadas de acuerdo con la siguiente tabla:

Horas empleadas	Factor de corrección C_t
Hasta 36 horas	1,00
Desde 36 a 72 horas	0,90
Desde 72 a 108 horas	0,80
Desde 180 a 144 horas	0,70
Desde 144 a 180 horas	0,65
Desde 180 a 360 horas	0,60
Desde 360 a 540 horas	0,55

Tabla P3. Factor de corrección según el COIT.

Como se puede observar el número de horas está comprendido en desde 180 a 360 horas, por lo que según la Tabla P3 el factor de corrección es de 0,60. Con ello, la ecuación del importe de horas de trabajo resulta de la siguiente forma:

$$H = 0{,}60 * 74{,}88 * 300 + 0{,}60 * 96{,}72 * 0 = 13.478{,}4€$$

Los honorarios totales por tiempo dedicado libres de impuestos ascienden a: *trece mil cuatrocientos setenta y ocho euros con cuarenta céntimos* (**13.478,4 €**).

4 Coste de redacción del Trabajo Fin de Grado

El importe de la redacción del Trabajo Fin de Grado se calcula de acuerdo a la siguiente expresión:

$$R = 0{,}07 * P * C_n$$

Donde:

- *P* es el presupuesto del Trabajo Fin de Grado
- C_n es el coeficiente de ponderación en función del presupuesto.

En la siguiente tabla se muestra el presupuesto calculado hasta el momento:

Recursos	Costes
Recursos hardware	886,49
Recursos software	1.477,64
Trabajo tarifado por tiempo empleado	13.478,4
TOTAL	**15.842,53**

Tabla P4. Presupuesto parcial

El presupuesto calculado hasta el momento asciende a 15.842,53 €. Como el coeficiente de ponderación para presupuestos menores de 30.050 € viene definido por COIT con un valor de 1,00 el coste derivado de la redacción del trabajo es de:

$$R = 0{,}07 * 15.842{,}53 * 1 = 1.108{,}97$$

El importe por redacción de trabajo asciende a la cantidad de: *mil ciento ocho euros con noventa y nueve céntimos* (**1.108,97 €**).

5 Material fungible

Materiales	Costes (€)
Folios	12
Tóner de impresora	5
Encuadernación	100
TOTAL	**117**

Tabla P5. Material Fungible.

6 Derechos de visado del COIT

Los gastos de visado del COIT, se tarifan mediante la siguiente ecuación:

$$V = 0{,}006 * P * C_V$$

Donde:

- *P* es el presupuesto del proyecto
- C_V es el coeficiente reductor en función del presupuesto del proyecto.

El presupuesto de ejecución material, calculado hasta el momento asciende a la cantidad de: 17.068,5 €.

El coeficiente C_V dado por el COIT para presupuestos menores de 30.050 € tiene el valor de 1,00 por lo que:

$$V = 0{,}006 * 17.068{,}5 * 1 = 102{,}4\,€$$

El coste de los derechos de visado del trabajo ascienden a la cantidad de: *ciento dos euros y cuarenta céntimos (***102,40 €**).

7 Gastos de tramitación y envío

Los gastos de tramitación y envío están fijados en 6,01 €.

8 Aplicación de impuestos

Para la actividad económica del presente trabajo el valor del Impuesto General de las Islas Canarias (I.G.I.C.) graba el presupuesto con un 7%. El coste total del trabajo con el I.G.I.C. incluido se desglosa en la siguiente tabla:

Descripción	Subtotal (€)
Recursos materiales	
Hardware	886,49
Software	1.477,64
Trabajo tarificado por tiempo empleado	13.478,4
Coste de redacción del trabajo	1.108,97
Material fungible	117
Derechos de visado del COIT	102,4
Gastos de tramitación y envío	6,01
Suma	17.176,91
IGIC (7%)	1.202,38
TOTAL	**18.379,29**

Tabla P6. Coste total del trabajo

El importe final al que asciende el presupuesto de este trabajo es de: *dieciocho mil trescientos setenta y nueve euros con veintinueve céntimos* (**18.379,29 €**).

Las Palmas de Gran Canaria a 9 de diciembre de 2014

Firma: Samuel Ángel Martel Santana