

Universidad de la República



Facultad de Ingeniería
Instituto de Computación
Montevideo – Uruguay

Proyecto de Grado

Aplicación de redes sociales a la educación a través de Internet.

Junio 2009

Alejandra Borghi
Yanet Cortés
Walter López Da Cunha

Tutores:
Martín Furno
Pablo Rebufello

Informe de Proyecto de Grado presentado al Tribunal Evaluador como requisito de graduación de la carrera Ingeniería en Computación.

Resumen

El presente proyecto consiste en la aplicación de nuevas tecnologías para el desarrollo de la educación a través de Internet. Su objetivo es brindar a la empresa *EviMed* nuevas formas de servicios educativos. La misma se encarga de ofrecer a sus afiliados (médicos) servicios que apuntan a la actualización profesional, área denominada Educación Médica Continua (EMC). Se intenta aportar soluciones que sigan las nuevas tendencias 'Web 2.0' para apoyar la propuesta educativa de una red social de la empresa que hace EMC por Internet llamada *eviDoctor*. La necesidad de fomentar la interrelación entre los médicos, la posibilidad de generar conocimiento en forma colectiva y el crecimiento de esta red, sirvieron como motivación para la realización de la primera parte del proyecto, la puesta en producción de una aplicación web colaborativa que puede ser editada por varios usuarios (Wiki) a la cual se denominó *WikiMed.* De esta manera se busca incentivar la participación de los usuarios, rompiendo con las barreras existentes entre médicos y máquinas, fortaleciendo el uso de las tecnologías de la información y la comunicación en la educación, no sólo en el ámbito de la medicina, sino en cualquier otro entorno educativo a través de Internet.

Durante el año 2008 y 2009 se llevaron a cabo experiencias con muy buenos resultados que evidencian la potencialidad de la 'inteligencia colectiva' como creadora de conocimiento. Actualmente *WikiMed* está en uso y disponible para los 3.500 médicos que conforman la red.

Una vez implantada la Wiki, se estudió la posibilidad de integrarla con el software de gestión de cursos *Moodle*, permitiendo la autenticación de usuarios en un único contexto, pero al encontrar un módulo ya existente que los integra, se dejó de lado esta alternativa.

Posteriormente, la Asociación de Psiquiatras del Interior (API), otra red afiliada a la empresa, manifestó una necesidad actual, compartir en tiempo real entre colegas sesiones con algunos de sus pacientes, viéndose dificultado esto por las distancias geográficas en la que se encuentran. Fue así que se desarrolló el sistema *G-Sell*. Se trata del desarrollo de una cámara de Gesell virtual (observar sin ser vistos) mediante transmisiones en vivo por Internet, además de crear una base de conocimiento apoyada en la información que se comparte y genera en estas sesiones, favoreciendo de esta forma a la Educación Médica Continua.

Esta aplicación diseñada como cliente inteligente que fue pensada inicialmente para el uso de un conjunto de médicos, ha sido implementada de manera generalizable, aplicando las nuevas tendencias 'Web 2.0'. Se presenta una solución que modela la realidad planteada permitiendo su aplicación no sólo en este entorno, sino en una amplia diversidad de disciplinas como la telemedicina, entornos de investigación y comerciales, entre otros, denotando el potencial e importancia académica que este proyecto tiene.

Además, se pretende lograr un producto final que sea amigable y de uso intuitivo para los usuarios, favoreciendo de esta forma su aceptación y utilización. El emisor es quien define una entrevista, le asocia la información relevante, emite un video con sonido incluido y autoriza a un grupo de usuarios receptores a visualizarla. Quienes a distancia sigan su desarrollo pueden incluso intercambiar mensajes a través del servicio de chat. Una vez culminada la sesión, el emisor podrá publicar las entrevistas, dejando estas a disposición para ser descargadas posteriormente previendo los casos de conexiones a Internet esporádicas.

El producto brinda a sus administradores la posibilidad de definir seguridad por usuario, proporcionando un marco de confidencialidad en los datos manejados. También se brinda la posibilidad de centralizar y compartir publicaciones, artículos o enlaces de interés para la rama de uso del sistema. En definitiva, todo aquello que lleve a interactuar en tiempo real rompiendo barreras geográficas, compartiendo información entre un grupo de usuarios, encontrará en este producto una herramienta para iniciar o fortalecer sus redes sociales.

Como resultado de ambas etapas, el proyecto concluyó con la creación de dos productos complementarios respecto a la experiencia que ambos aportan. En el caso de *WikiMed*, se trata de un producto originado a partir de herramientas de 'software libre', configurado para un uso particular y puesto en producción en un ambiente real. Mientras que el sistema *G-Sell*, se trata de un desarrollo a medida que permitió incorporar la componente de desarrollo ausente en *WikiMed*.

Palabras claves: Educación Médica Continua, Cámara de Gesell, Interacción hombre máquina, Inteligencia Colectiva, Telemedicina, Psiquiatría, Redes sociales, Web 2.0, Web 3.0, Intervenciones educativas en línea.

Agradecimientos

Al ingeniero Antonio López, representante de la empresa *EviMed,* quien ha sido un guía indiscutible en todo el proyecto, junto con la visión del doctor Álvaro Margolis y el ingeniero Marcelo Bondarenco.

A Elisa Martínez, licenciada en comunicación, integrante del equipo *EviMed,* quien fue nuestro nexo directo con los usuarios en el uso de la Wiki desarrollada, además de aportar siempre su visión innovadora sobre la comunicación entre los usuarios de la red.

A los ingenieros Martín Furno y Pablo Rebufello por su labor como representantes y tutores del presente trabajo.

Al equipo de d2B Network, empresa de desarrollo de soluciones digitales [6], que nos proporcionaron su visión y experiencia respecto a la tecnología Microsoft, desarrollo en lenguaje C sharp y al uso de Windows Presentation Foundation (WPF), óptima para nuestro enfoque.

Al Centro de Posgrados y Actualización Profesional (CPAP) del Instituto de Computación de la Facultad de Ingeniería de la Universidad de la República por posibilitar que todos los integrantes de este grupo pudieran participar del curso de actualización Aplicaciones web avanzadas (Web 2.0).

Al señor Pablo García, gerente de Microsoft, por disipar dudas respecto a los productos Microsoft utilizados.

Por último, pero no por eso menos importante, al apoyo incondicional de nuestras familias.

Índice de contenidos

SECCIÓN I

1. Introducción....2
1.1 Motivación....2
1.2 Objetivos....3
1.3 Metodología de trabajo....4
1.4 Resultados esperados....4
1.5 Conclusiones....5
1.6 Organización del trabajo....6
1.7 Cronograma....7

SECCIÓN II

Parte A – WIKIMED

2. Desarrollo de la investigación....9
2.1 Marco teórico....9
2.2 Estado del arte....10
3. Alcance....12
4. Implementación....13
4.1 Diseño de interfaces....13
4.2 Ajustes....14
4.2.1 Permisos....14
4.2.2 Inicio....14
4.2.3 Artículos....15
4.3 Decisiones tomadas....17
5. Pruebas y resultados....18
5.1 Pruebas....18
5.2 Resultados....19

Parte B - CÁMARA DE GESELL

2. Desarrollo de la investigación....21
2.1 Marco teórico....21
2.2 Estado del arte....22
3. Alcance....28
4. Implementación....30
4.1 Descripción del entorno de desarrollo....30
4.1.1 Cliente....31
4.1.2 Servidor....31
4.2 Arquitectura....31
4.2.1 Gestión de base de datos....33
4.2.2 Capa de acceso a datos....33
4.2.3 Chat....34
4.2.4 Emisión....35
4.2.5 Publicación y descarga de archivos....36
4.3 Diseño....38
4.3.1 Servidor....38
4.3.2 Cliente....43
4.4 Requerimientos....48
4.4.1 Servidor....48
4.4.2 Cliente....50
4.5 Solución tecnológica....51
4.5.1 Windows Presentation Foundation (WPF)....51
4.5.2 Windows Communication Foundation (WCF)....51
4.6 Implementación....51
4.6.1 Implementación de acceso a datos....52
4.6.2 Desarrollo de la presentación....53

4.6.3 Desarrollo de captura y streaming de multimedia.....55
4.6.4 Desarrollo de chat.....56
4.6.5 Desarrollo de publicación y descarga.....57
4.6.6 Desarrollo de funcionalidades complementarias.....58
4.7 Dificultades.....58
4.8 Decisiones tomadas.....59
5. Pruebas y resultados.....60
5.1 Pruebas.....60
5.1.1 Pruebas realizadas al servicio de chat.....60
5.1.2 Pruebas realizadas al consumo por demanda.....61
5.1.3 Pruebas realizadas sobre video.....61
5.1.4 Evaluación de performance para emisión.....62
5.1.5 Pruebas sobre acceso a datos.....63
5.1.6 Pruebas sobre la presentación.....63
5.2 Resultados.....65

SECCIÓN III

6. Conclusiones y trabajo futuro.....67
6.1 Conclusiones.....67
6.2 Trabajo futuro.....68
6.3 Aprendizaje.....70
Referencias.....72
Glosario.....74
Índice de ilustraciones.....79

INFORME FINAL

1. Introducción

Posiblemente Marshall McLuhan (1911 – 1980) haya sido el primero en imaginarse una 'aldea global' [1]. Debido al progreso tecnológico, todos los habitantes del planeta empezarían a conocerse unos a otros y a comunicarse de manera instantánea y directa.

Al menos en la teoría, el principio que impera en este concepto es el de un mundo interrelacionado, con estrechez de vínculos económicos, políticos y sociales, producto de las tecnologías de la información y la comunicación, particularmente Internet. Esta profunda interrelación entre todas las regiones del mundo originaría una poderosa red de dependencias mutuas.

Como paradigma de aldea global, McLuhan elige la televisión, un medio de comunicación de masas a nivel internacional, que en esa época empezaba a ser vía satélite. En efecto, cuando McLuhan murió, la televisión por cable, y ni hablar de Internet, todavía no eran una realidad mundial. Los habitantes de la 'aldea global' aún poco sabían sobre interactividad, e-books, multimedia y videoconferencias. Pero la obra de este sociólogo canadiense ha dejado un marco teórico que permite estudiar y comprender la naturaleza de estos nuevos medios que han revolucionado la historia de la comunicación de la humanidad.

Años más tarde, en la trilogía 'La Era de la Información' [2] el sociólogo español Manuel Castells, analiza la gestación de lo que se conoce como 'revolución tecnológica'. La sociedad informacional se orienta hacia la acumulación de conocimiento elevando los grados de complejidad en el procesamiento de la información. La superación permanente de las tecnologías de la información y la comunicación genera un círculo de interacción de las fuentes del conocimiento con la aplicación de estas tecnologías, revolucionando continuamente los procesos de generación y distribución de la propia información y conocimiento.

El presente documento corresponde al proyecto: "Redes Sociales y la Web Semántica (Web 3.0): estado del arte y aplicaciones a la educación a distancia", de la asignatura Proyecto de Grado de la carrera Ingeniería en Computación de la Facultad de Ingeniería.

El proyecto fue propuesto por la empresa *EviMed* [3], encargada de ofrecer a sus usuarios, médicos uruguayos y de la región, servicios que apuntan a la Educación Médica Continua [4]. Se buscaba incrementar los servicios educativos y facilitar la interacción entre los médicos que conforman la red social *eviDoctor* [5]. Para alcanzar este objetivo, se esperaba poder aplicar las nuevas tendencias de Internet; redes sociales, también llamada 'Web 2.0' y redes semánticas, definidas como 'Web 3.0'.

El título original del proyecto fue cambiado por "Aplicación de redes sociales a la educación a través de Internet". Una vez realizado el estudio del estado del arte y de conocer a fondo qué son realmente las tendencias Web 2.0 [ver anexo A1-2. Tendencia Web 2.0], y Web 3.0, [ver anexo A1-1. Web Semántica]; teniendo en mente las necesidades planteadas por *EviMed* y aprovechando la posibilidad de usar la red social *eviDoctor* como prueba de campo, se optó por buscar soluciones que apuntan a la tendencia Web 2.0. La Web Semántica no se aplica en el entorno de trabajo en el que se desarrolló este proyecto, por esto se decidió poner un título que fuera más representativo de lo que es el trabajo final.

1.1 Motivación

La empresa *EviMed* cuenta con su propia plataforma educativa llamada *eviDoctor* que se utiliza para ofrecer un servicio de Educación Médica Continua realizada mediante 'boletines', 'videos', 'encuestas' y el 'ateneo electrónico', con casos clínicos para discutir. Actualmente, más de 3.500 médicos de todo el país reciben semanalmente esta información y conforman así, la red social *eviDoctor.* La necesidad de fomentar la interrelación entre los médicos, la posibilidad de generar conocimiento en forma colectiva y el crecimiento de esta red, sirvieron como motivación para la realización de la primera parte del proyecto: la puesta en producción de *WikiMed* (la Wiki de *EviMed*) [6].

En la segunda parte del proyecto se toman en cuenta las necesidades de la Asociación de Psiquiatras del Interior, quienes buscan romper las barreras geográficas que separan a sus

miembros. Estos profesionales se encuentran dispersos en todo el país, por lo que juntar un grupo de médicos para poder compartir físicamente una entrevista, lleva asociado un gran esfuerzo de organización, logística y traslado de personas. Esto, además de no haber encontrado una herramienta adecuada que cumpliera con todas las expectativas planteadas, motiva a crear una solución (*G-Sell*) que permita compartir información y entrevistas en vivo entre médicos que se encuentran distantes, sin necesidad de movilizar personas.

Además, en algunos casos, los usuarios no cuentan con conexión permanente a Internet. Solamente tienen disponible una conexión en forma regular por un tiempo acotado. Esto motiva a encontrar una solución que les permita consumir contenido en forma local luego de haber descargado la información a través de Internet o de generar el contenido también en forma local para subirlo cuando se cuente con una conexión disponible.

1.2 Objetivos

Los objetivos principales que inicialmente fueron presentados en este proyecto se pueden describir de la siguiente manera.

- Enriquecer la forma de entrega de los materiales que se producen en el marco del servicio de Educación Médica Continua, soluciones que apoyen a la propuesta educativa de *eviDoctor*, así como facilitar la interacción entre los médicos que conforman la red social, a través de los conceptos conocidos como Web 2.0. Esto incluye multimedia, herramientas de colaboración, podcasts y wikis.
- Construir un servicio web que facilite la interacción de los médicos con el sistema y que los motive a participar en las actividades a distancia.
- Realizar una publicación científica con el resultado del trabajo realizado para ser presentado en algún congreso internacional.

Durante el transcurso del trabajo y en respuesta a las diferentes necesidades surgidas, dichas metas fueron replanteadas para cumplir los desafíos establecidos.

- Generar nuevas soluciones que apoyen a la propuesta educativa de *eviDoctor*, incorporándolas a los servicios actuales. Se espera que las soluciones sigan las nuevas tendencias 'Web 2.0' (redes sociales). Asi se intenta fortalecer el uso de las tecnologías de la información y la comunicación en la educación, no sólo en el ámbito de la medicina, sino en cualquier otro entorno educativo.
- Proporcionar el apoyo y soporte necesario para realizar una publicación científica que trate sobre esta temática a ser presentada en un congreso internacional ya que la integración de las tecnologías es actualmente un tema de primer orden en el ámbito de la Educación Médica Continua.
- Bajo las condiciones expresadas por *EviMed* acerca de las necesidades planteadas por la Asociación de Psiquiatras del Interior (API) se debe crear un sistema que implemente en forma virtual a través de Internet las funciones de un dispositivo que conocen y aprecian en su práctica clínica: la Cámara de Gesell
    - El servicio debe estar enmarcado en un portal de contenidos.
    - Las entrevistas podrán ser vistas en vivo y posteriormente descargadas 'a demanda' para su uso en forma desconectada.
    - La versión 'en vivo' debe permitir a los asistentes remotos la posibilidad de hacer preguntas a través de un chat.
    - Se debe permitir la posibilidad de crear grupos de afinidad (por ejemplo Psiquiatras, Psicólogos, etc.).
    - Se debe contar con cierto grado de seguridad de manera de proteger los datos de las entrevistas por tratarse de información confidencial.
    - Los usuarios del sistema deben poder compartir novedades, artículos o links de interés y publicaciones.

1.3 Metodología de trabajo

Durante el transcurso del proyecto se utilizaron dos diferentes enfoques respecto a la metodología de trabajo para resolver las problemáticas planteadas en cada caso. A partir de ambos estados del arte, en *WikiMed* se imponía la aplicación de la filosofía de 'software libre' [7], mientras que para el caso de *G-Sell* fue necesario realizar un desarrollo a medida pasando por las distintas etapas de su ciclo de vida.

Para la puesta en funcionamiento de la Wiki se siguió la dinámica que marca en primera instancia una selección de la herramienta necesaria, para su posterior configuración e implantación en el ambiente de producción.



Figura 1.1: Enfoque orientado a software libre usado en WikiMed.

Al realizar el desarrollo a medida, en cambio, se aplicó el modelo cascada realimentado [8]. Esta es una variante al modelo de cascada puro que permite interacción entre las distintas fases del ciclo de vida del software. La visión del modelo dice que el desarrollo de software puede ser a través de una secuencia simple de fases. Cada fase tiene un conjunto de metas bien definidas y las actividades dentro de una fase contribuyen con su satisfacción.

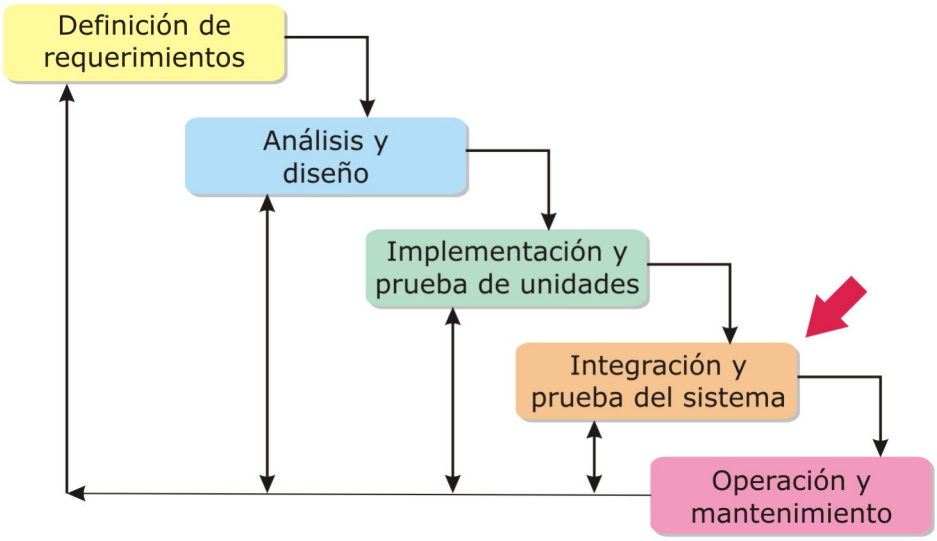


Figura 1.2: Modelo cascada realimentado utilizado en el desarrollo de G-Sell.

En la figura se esquematiza el modelo seguido donde las fechas muestran el flujo de información y secuencialidad entre las distintas fases. En este caso las flechas de avance representan el flujo normal, mientras que las dirigidas hacia atrás la realimentación para una nueva iteración.

Aplicado al desarrollo de *G-Sell* se alimentó el modelo con un paso inicial donde se definió un prototipo en papel, [ver anexo A5-2. Bosquejo de interfaces en papel], a partir del cual se comenzó el trabajo. El desarrollo se detuvo en la primera iteración, como marca la flecha en la figura, justo antes de poner el producto en producción.

1.4 Resultados esperados

Se pretende crear un producto aplicando las nuevas tendencias de Internet y aprovechando la evolución que ha tenido en los últimos años, sin perder de vista el cometido final de promover el aprendizaje colaborativo. Se espera que este producto aporte nuevas

funcionalidades a la plataforma *eviDoctor* y que pueda quedar disponible en producción al finalizar el proyecto.

Se busca incentivar la participación de los usuarios, en el marco del sistema de Educación Médica Continua, rompiendo con las barreras existentes entre médicos y máquinas. Se debe tener en cuenta que los médicos no son muy afines al uso de equipos informáticos, prefiriendo la comunicación personal.

En lo que refiere al sistema *G-Sell*, se espera a través del mismo, enriquecer sustancialmente la interacción entre sus usuarios. Las entrevistas, que son las unidades de trabajo de la aplicación, permiten que varios usuarios coincidan en un mismo lugar virtual para una emisión en vivo. En esta instancia los usuarios pueden compartir sus conocimientos e intercambiar ideas sin tener que estar físicamente juntos, con las ventajas que esto implica.

En este contexto se busca generar conocimiento colectivo asociado a cada entrevista fuera del entorno de la emisión en vivo. Esto se logra a partir de la posibilidad de definir la metadata para los datos de una entrevista. Como los datos preceden y anteceden a cada emisión en vivo, se brinda la posibilidad a los usuarios de sincronizar las entrevistas descargándolas localmente y así consumir la información en cualquier forma y lugar, independientemente de que tenga o no conexión a Internet.

Además, se pretende lograr un producto haciendo hincapié en que la interfaz de usuario sea amigable e intuitiva, aplicándose el concepto de 'diseño web centrado en el usuario'. Para favorecer su aceptación y utilización se toman en cuenta, entre otras características:

- Botones que ilustren las funciones a ejecutar, como pueden ser comenzar emisión, detener emisión o descargar videos emitidos.
- Formularios parametrizados que permitan el ingreso de información, según la especialidad seleccionada.
- Listas desplegables que faciliten la carga de los datos.
- Imágenes preliminares de videos disponibles para su descarga, facilitando la rápida selección.
- Incorporación de paginado para la selección de publicaciones.

1.5 Conclusiones

La primera parte del proyecto concluye con la puesta en funcionamiento de una Wiki. Esta es una nueva funcionalidad que se incorpora para el uso de la red social *eviDoctor*. La Wiki se encuentra actualmente en producción y su uso está disponible para aproximadamente 3.500 médicos que conforman la red. Fue puesta en marcha como apoyo en la generación de conocimiento colectivo en cursos realizados durante 2008 y 2009. Quienes efectivamente participaron de la generación de contenido en *WikiMed* fueron los médicos y otros profesionales de la salud que estaban inscriptos en dichos cursos.

Como producto de esta innovación se pudo observar el inicio del trabajo colaborativo entre médicos, con el aporte de muchos de ellos en la edición de artículos, y el consumo de otros que aún no se han animado a participar como editores. También se puede observar la calidad de la información generada, avalada por la profesionalidad del tipo de usuario que puede modificar su contenido. *WikiMed* es un claro ejemplo de la aplicación de las nuevas tendencias Web 2.0, funcionando como infraestructura para la generación de contenido colectivo de una red social, en este caso, médica.

En la segunda parte del proyecto se construye un sistema desarrollado a medida, por la manera en que fue diseñado se obtiene una herramienta que puede ser usada en cualquier disciplina y para cualquier tipo de información. En particular, para el caso de la Asociación de Psiquiatras del Interior, el poder usar una herramienta de estas características podría significar para los médicos un mayor intercambio de situaciones con sus pacientes y de conocimientos sobre los temas abordados. La información generada podría ser aplicada también como referente en otras unidades de trabajo.

La experiencia del equipo de comunicación de la empresa conduce a asegurar que los médicos mayoritariamente no son el tipo de personas que priorizan el uso de sistemas informáticos, ya que no están incorporados a su práctica clínica habitual (salvo excepciones). Por esta razón se hizo hincapié en motivarlos para poder generar conocimiento colectivo y hacerlos partícipes del uso de las herramientas.

Como resultado de ambas etapas, el proyecto finalmente concluyó con la creación de dos productos, cumpliéndose de esa forma con todos los objetivos propuestos inicialmente. En este punto se considera importante destacar el hecho de que ambas realizaciones se complementan respecto a la experiencia que aporta cada una. En el caso de *WikiMed*, se trata de un producto originado a partir de una herramienta de software libre, configurado para un uso particular y puesto en producción en un ambiente real, con todas las implicancias que esta tarea conlleva. Mientras que el sistema *G-Sell*, un acercamiento a una Cámara de Gesell virtual que permite además, la gestión de entrevistas y su reproducción a demanda fuera de línea, no llegó a la fase de puesta en producción aunque permitió incorporar la componente de desarrollo ausente en *WikiMed*. Es por este motivo que se presentan ambos productos en forma separada.

Además, se ganó experiencia y conocimiento en la aplicación de nuevas herramientas, tecnologías y lenguajes. Entre las herramientas más importantes se encuentran *MediaWiki* [9]; Visual Studio con sus diferentes propuestas a saber, Windows Presentation Foundation (WPF), Windows Communication Foundation (WCF), Linq to Sql y Sql Server. Se incursionó en forma reducida en el uso del lenguaje PHP para configuraciones y tareas puntuales de la Wiki. Se obtuvieron conocimientos sobre el lenguaje C sharp a través de Visual Studio.

1.6 Organización del trabajo

El trabajo está dividido en tres documentos que son presentados en forma separada. El primero corresponde al informe final, que es la sección principal de la entrega y describe en detalle los objetivos, metodologías, resultados, conclusiones y desarrollo del proyecto. En un segundo documento se entrega un compendio de anexos que permiten exponer con más detalle, por su importancia, algunos de los temas abordados. El tercer y último documento presentado contiene el manual de usuario, describiendo la forma en que se puede acceder a las distintas funcionalidades de la aplicación *G-Sell.*

El presente documento, que corresponde al informe final, está estructurado en tres secciones.

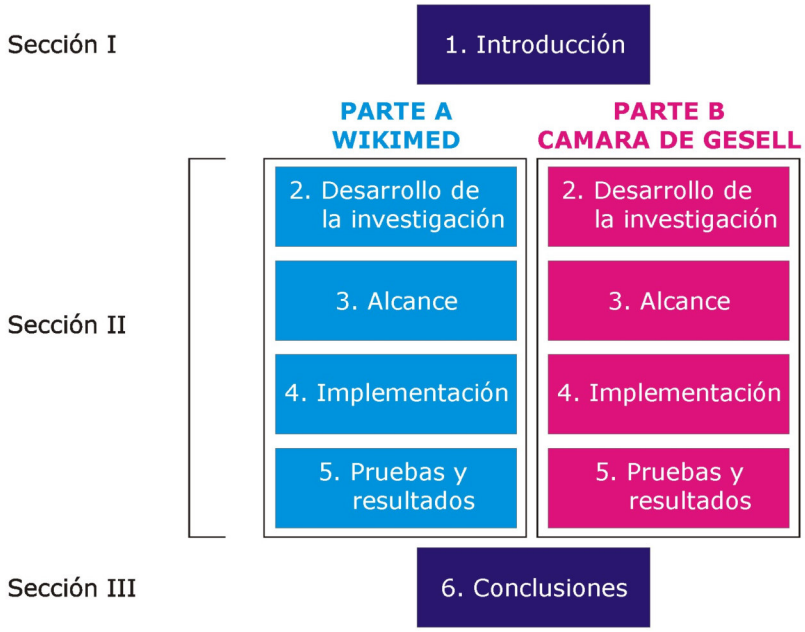


Figura 1.3: Organización del documento.

En la primera se le presenta al lector una introducción al proyecto familiarizándolo en las temáticas abordadas, conceptos y términos usados en todo su contenido.

Posteriormente, se divide la segunda sección en dos partes con enfoques similares, pero donde cada una corresponde al producto creado:

- Parte A – WIKIMED
  Selección, configuración y puesta en producción de *WikiMed*.
- Parte B – CÁMARA DE GESELL
  Especificación, diseño e implementación de *G-Sell*.

Dentro de cada parte se describe el desarrollo de la investigación detallando el marco teórico y realizando un estudio del estado del arte. A continuación se define el alcance del proyecto y se describe la arquitectura, diseño, implementación, desarrollo, pruebas, resultados, así como la puesta en funcionamiento de cada uno de los productos.

Como cierre, en la tercera sección del documento se presentan las conclusiones del trabajo, el alcance logrado, el aprendizaje obtenido, además de las funcionalidades pendientes a ser tenidas en cuenta en futuras versiones.

1.7 Cronograma

En la siguiente imagen se tiene una comparativa entre el cronograma planificado inicialmente y el cronograma realizado durante el proyecto. Se visualiza en color verde el desarrollo en particular del producto *WikiMed*, en azul el de *G-Sell* y en violeta la parte del proyecto genérica.

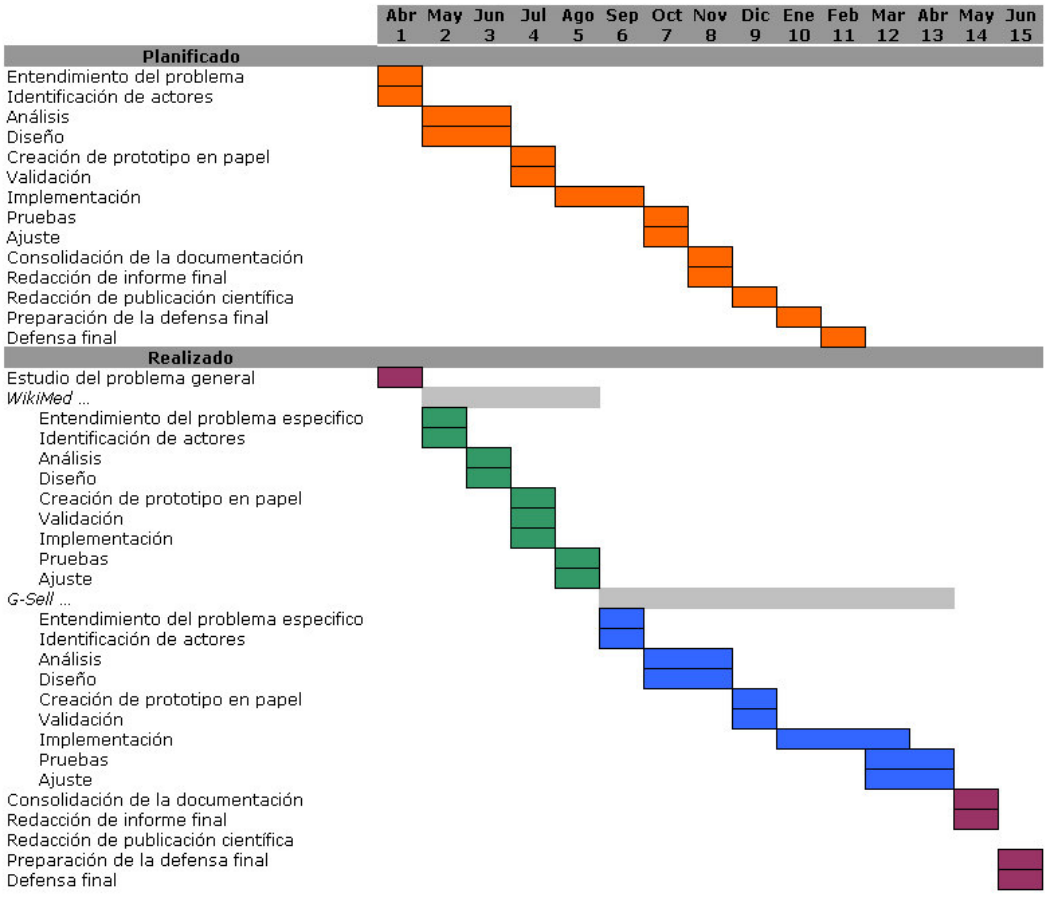
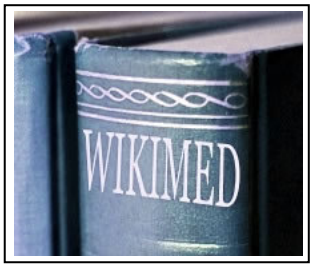


Figura 1.4: Cronograma comparativo.

Parte A – WIKIMED



2. Desarrollo de la investigación

Se comenzó con la investigación de las nuevas tendencias Web, la evolución de Internet en los últimos años y posibles herramientas de desarrollo que permitieran implementar soluciones de acuerdo a estas tendencias.

Se hizo una selección de las herramientas que mejor se aplicaban al proyecto, teniendo en mente el objetivo de fomentar la posibilidad de compartir el conocimiento y la interactividad entre los usuarios apostando siempre a la 'Educación a Distancia' [10].

Se investigaron alternativas que permitieran apoyar los sistemas de educación a distancia y a su vez sirvieran de plataforma para cumplir los propósitos del proyecto.

- El sistema iTunes University, aportado y mantenido por la empresa Apple, fue una de ellas. En esencia esta solución brinda una forma diferente de potenciar los sistemas de enseñanza, información educativa, aprendizaje, investigación y conocimientos que las universidades y otros centros educativos aportan a la comunidad [ver anexo A1-4. Estudio de iTunes U]. Esta opción fue descartada ante restricciones encontradas en su análisis.
- Genexus Rocha es una herramienta de desarrollo de software, cuya última versión apunta a mejorar la calidad de aplicaciones generadas para Web 2.0. En este marco, se buscó aplicarla enfocada a la implementación de la Wiki [ver anexo A1-3. Funcionalidades Web 2.0 de Genexus].

2.1 Marco teórico

Las Wikis comparten numerosas características con los blogs, pero representan un paso más en el acelerado e imparable proceso de socialización de la red. La palabra 'wiki-wiki' significa en hawaiano 'rápido', expresión repetida entre los remeros de las canoas que conocen como nadie el valor del esfuerzo colaborativo en la consecución de una meta. El término se utiliza entre los internautas para designar una amplia colección de páginas web que pueden ser visitadas y editadas libremente por cualquier individuo. El sitio crece gracias al perpetuo trabajo de una comunidad de usuarios interesada en los mismos temas [11].

En 1995 Ward Cunningham creó la primera Wiki [12], una enciclopedia sobre programas de computación que, con el tiempo, se convertiría en un sitio casi de culto sobre programación extrema (extreme programming). Sin embargo, la mayor e indiscutible fuente de conocimiento en la red es la Wikipedia [13], un compendio del conocimiento humano en permanente proceso de construcción y en el que participan a diario cientos de miles de usuarios en más de un centenar de idiomas. Se ha convertido, por derecho propio, en la mejor enciclopedia de código abierto de Internet. Actualmente existe una enorme variedad de proyectos Wiki gestionados por comunidades de usuarios que comparten algún interés común y se han acuñado términos como: 'wikidictionary', 'wikibook', 'wikitravel'. Desde un punto de vista estructural, una plataforma tipo Wiki contiene la información (organizada, como en todo proyecto web, en formato hipertexto), en una base de datos, vinculada a un sistema para gestión de contenido CMS (por su sigla en inglés), que genera las páginas de manera dinámica.

Una publicación realizada por la prestigiosa revista científica Nature [14] dio a conocer un estudio según el cual la calidad de la información incluida en la Enciclopedia Británica [15] y en Wikipedia es similar. El artículo se titula "Internet encyclopaedias go head to head", escrito por Jim Giles [16] y publicado en la revista en diciembre de 2005. Debe tenerse en cuenta que para que un artículo sea publicado en esta revista arbitrada tiene que haber pasado varias aprobaciones que en cierta forma aportan un grado de exactitud a su contenido. Por la relevancia de este artículo en base a los resultados obtenidos se ha decidido incluir una copia del mismo como anexo [ver anexo A5-3. Internet encyclopaedias go head to head]. Dicho artículo ha sido referenciado por innumerables sitios de Internet cuando se discuten puntos a favor y en contra de la generación de conocimiento a través de las wikis y la validez del producto obtenido por la inteligencia colectiva.

En este estudio se compararon 42 entradas de artículos de ambas enciclopedias. El estudio estuvo a cargo de expertos que desconocían de cuál de las dos enciclopedias provenía la información presentada. El ejercicio reveló numerosos errores en ambos casos, pero en proporciones similares. El promedio de entradas de 'ciencia' que figura en Wikipedia contiene un entorno de cuatro inexactitudes mientras que en la Enciclopedia Británica alrededor de tres. También se realizó una elección de entradas de los sitios web de Wikipedia y la Enciclopedia Británica en una amplia gama de disciplinas científicas, que fueron enviadas a expertos para su revisión, con resultados similares.

2.2 Estado del arte

La investigación de las nuevas tendencias abrió un amplio espectro de alternativas que podían ser implantadas para cumplir con el objetivo [ver anexo A1-2. Tendencia Web 2.0]. En vista del estudio comparativo sobre las wikis, publicado por Giles, en busca de herramientas que permitan la creación de conocimiento en forma colaborativa y la necesidad planteada por la empresa *EviMed* de nuevas formas para fomentar la participación de los usuarios en el entorno de la Educación Médica Continua, se decide aprovechar la existencia de la red social *eviDoctor* para implantar una solución haciendo uso de la actual infraestructura de *EviMed*. El cometido es poner en producción una herramienta 'open source' (código abierto) en un ambiente real. Es así que surge la creación de *WikiMed*, buscando realizar pruebas de campo con la herramienta investigada.

Existen varias alternativas "open source", generalmente programadas en Perl o PHP, que implementan una Wiki. Estas frecuentemente suelen utilizar una base de datos para almacenar la información introducida. Entre las más usadas actualmente se pueden destacar:

- DokuWiki [17]: Una Wiki completa, escrita en PHP y sin necesidad de base de datos (usa solamente archivos de texto).
- MediaWiki: Utilizada en todos los proyectos de la fundación WikiMedia [18], está basada en PHP y MySQL.
- OpenWikiNG [19]: Programada en ASP para alojarse en plataformas Windows.
- TikiWiki [20]: CMS completo que incluye una Wiki programada en PHP y MySQL.

	MediaWiki	TikiWiki	DokuWiki	OpenWikiNG
Libre y open source	✓	✓	✓	✓
Lenguaje de programación	PHP	PHP	PHP	ASP, XML, XSL,CSS
Sistema operativo	*nix, Windows, Mac OS X	Linux, FreeBSD, Windows, Mac OS X	Unix, Windows, Mac OS X	Windows
Servidor Web	Cualquiera con soporte PHP	Cualquiera con soporte PHP	Cualquiera con soporte PHP	Cualquiera con soporte PHP
Soporte a datos				
MySQL	✓	✓	✗	✓
PostgreSQL	✓	✓	✗	✓
SQLite	✓	✓	✗	✗
Seguridad				
Encriptación de mails	✓ (plug-in)	✓	✓ (opcional)	✗
Listas negras	✓	✗	✓ (opcional)	✓
Permisos en páginas	✓	✓	✓	✓
Autenticación	✓	LDAP, Vpopmail, POP3, IMAP, Basic HTTP y otros	LDAP, MySQL, PostgreSQL	Windows
Estadísticas				
Cambios recientes	✓	✓	✓	✓
Páginas buscadas	✓	✓	✓ (plug-in)	✗
Páginas huérfanas	✓	✓	✓ (plug-in)	✗
Páginas más o menos populares	✓	✓	✗	-
Visitantes recientes	✓ (plug-in)	✓	✗	✗
Análisis	✓ (opcional)	✗	✓ (plug-in)	✗

Figura A2.1: Comparativa entre principales alternativas para Wiki.

En la tabla, obtenida a partir de la aplicación de libre acceso WikiMatrix [21], se puede apreciar un resumen comparativo de características entre las principales herramientas existentes, estables y ampliamente utilizadas para instalar una Wiki.

La Wiki más importante que existe en la actualidad es la versión en inglés de Wikipedia, aunque existen otras ajenas a este proyecto, estas cuentan con una menor participación de usuarios, generalmente debido al hecho de ser mucho más especializadas.

Con los resultados comparativos a la vista y considerando que la alternativa creada por la fundación WikiMedia (soporte de Wikipedia) es un estándar de facto se optó por implantar el paquete de software *MediaWiki*.

Se incursionó también en la investigación de un sistema de gestión de contenidos llamado *Moodle* [22] y su posible integración con *WikiMed*. Esta herramienta es un software libre y de código abierto que permite a los educadores crear y gestionar comunidades de aprendizaje en línea a través de Internet. Por medio de este sistema es posible desarrollar actividades formativas y de comunicación tales como chats, foros, mensajería interna, encuestas, talleres, wikis, diarios y foros, entre otros. Los creadores de Moodle afirman que su solución es fácil de integrar con varias actividades generadas por distintas aplicaciones educativas, en su gran mayoría de software libre, entre las que se pueden destacar JClic, YouTube, Gimpes, Audacity, eXe, Link, Videotutoriales, Hot Potatoes y openOffice.org 2.0 [22].

Ante el requerimiento de integración entre *Moodle* y *MediaWiki* se estudió la posibilidad de que los usuarios de ambas herramientas pudieran autenticarse en un único contexto. En este caso se encontró el módulo MediaWikiSSO [23] con un nivel de desarrollo avanzado que soluciona eficazmente este problema.

3. Alcance

Si bien el proyecto siempre tuvo claro su enfoque, no tenía tan definido su alcance. Se comenzó con la Wiki, donde se aplicaron algunos principios de la Web 2.0. Su aporte fue importante para los médicos, poniéndose en producción de inmediato en cursos virtuales realizados en junio, agosto y octubre de 2008.

El aumento participativo de los médicos, la posibilidad de compartir conocimiento desarrollando discusiones e intercambiando opiniones sobre casos clínicos planteados, y la interactividad entre ellos, fue el fiel reflejo de la utilidad de *WikiMed*.

La creación de la Wiki fue el primer intento de cumplir con los objetivos del proyecto, formó parte de la propuesta educativa de *eviDoctor* incorporándose a sus servicios actuales. Su aplicación consiste en la creación de un artículo basado en la exposición de un caso como una 'consigna', por ejemplo 'Estrategias para mejorar la comunicación entre los integrantes del equipo de salud, ante el paciente con cáncer' y se comienza a desarrollar sobre este tema aportando conocimiento individual a la tarea colectiva. Es de esa forma que los usuarios participantes van construyendo el artículo.

Solamente podrán participar de la construcción de un artículo aquellos participantes 'habilitados' por un usuario con perfil de 'administrador'. Este último es el encargado de organizar la información contribuyendo, de esta forma, a la creación definitiva del contenido.

El aporte de *WikiMed* y los resultados obtenidos se ven reflejados en la creación de dos artículos [ver anexo A5-1. Experiencia en Wikimed]. Esto último motiva el desarrollo de una publicación científica que, si bien aún no ha concluido, será tema de interés en el ámbito de la Educación Médica Continua.

4. Implementación

Para alcanzar el objetivo de incorporar una plataforma donde los usuarios pudieran generar y compartir contenido en forma grupal, se instaló la solución de libre distribución *MediaWiki*. Esta herramienta implementada en PHP y configurada con una base de datos MySQL, fue personalizada según los requerimientos planteados por el equipo de *EviMed*, permitiendo de esta forma su utilización en cursos virtuales que la empresa realiza periódicamente.

Para la creación de *WikiMed*, se trabajó en dos aspectos que permiten un uso intuitivo y una integración sencilla al actual entorno de trabajo.

- Se hizo hincapié en la interfaz de usuario, personalizándola a las particularidades de la empresa y limitando algunas funcionalidades estándares en una Wiki.
- Se realizó la migración de información de los usuarios existentes en la base de datos de *EviMed* a la base de datos que maneja *MediaWiki*, depurando dicha información y definiendo roles de administrador o editor.

4.1 Diseño de interfaces

El diseño se enfocó principalmente en la interfaz de usuario, que debería ser presentada de forma tal que motivara a los usuarios a consumirla, ya sea para quienes sabían de lo que se trataba una Wiki, como para los que nunca se habían enfrentado a una. El hecho de tener un entorno simple y amigable estaba especialmente pensado para motivar su uso y generar así el conocimiento colectivo buscado.

En el diseño de *WikiMed* se aplicaron algunos patrones básicos de la Web 2.0 [ver anexo A2-1. Ocho patrones básicos para el desarrollo de la Web 2.0].

1- Aprovechar la inteligencia colectiva.
Los usuarios participan explícitamente en la generación de contenidos dispuestos en *artículos* y expuestos en la Wiki. Se confía en ellos para su creación y se ve claramente el efecto de red ya que cuantos más usuarios participan, más se enriquecen los conocimientos con nuevos aportes a los casos expuestos. Al personalizar la Wiki se les proporciona un marco amplio para la creación de contenidos y búsquedas sobre ella.

2- Los datos son el nuevo "Intel Inside".
La creación de los datos en este ámbito posee un gran valor por la manera en que estos se generan. Se necesita de ciertos permisos de acceso y controles para su creación. Esto lleva a la creación de una base de datos difícil de recrear compuesta por información confidencial por tratarse de temas médicos. La empresa pasa a ser dueña de esos datos por lo que tiene la autoridad de hacer que algunos derechos sean reservados si así lo desea. Los datos generados en la Wiki pueden ser reutilizados por otras aplicaciones de la empresa, como por ejemplo boletines o ateneo electrónico, generando de esa forma mecanismos de sindicación y distribución.

3- Innovar en el ensamble.
WikiMed se trata de una herramienta que facilita la creación de datos y servicios, que mediante el aumento del contenido generado por los usuarios provee un modelo de crecimiento escalable. Es una herramienta que, al igual que blogs, foros y listas de correo, ayuda a favorecer el desarrollo.

4- Experiencia rica de usuario.
La Wiki es un ejemplo de lo que actualmente se denominan aplicaciones livianas. No es necesaria la instalación de aplicaciones en cada máquina para la discusión de un *artículo* dado que se puede acceder a éste en línea. Cuenta con su propio editor para modificar o comentar el artículo, resultando algo muy simple y fácil de usar. El usuario 'administrador' de la Wiki puede personalizarla y adaptarla al enfoque que

considere conveniente, basándose en el comportamiento de los usuarios. Es una aplicación que aumenta el trabajo colaborativo, lo que se considera una de las buenas prácticas que menciona este patrón.

5- Software por sobre el nivel de un único hardware.
No se aplica este patrón.

6- Beta perpetuo.
Los usuarios participan en todo momento y son ellos quienes a partir de la exposición de un artículo, lo discuten construyéndolo a su vez. Se trata de una herramienta donde el usuario es el protagonista, siendo implícitamente su principal tester al discutir el contenido que en esta se genera. La Wiki es un beta perpetuo, ya que debido al surgimiento de nuevas versiones, la modificación de su configuración, o instalación de plugins se le agregan nuevas funcionalidades, resultando invisibles a los ojos del usuario final. Se protege la privacidad de la información, dado que no todos los usuarios 'clientes' de la empresa tienen acceso a la discusión de un determinado artículo. Existen políticas de seguridad y privacidad que el usuario administrador de la Wiki manipula.

7- Aprovechar la larga cola.
Se refiere a sacar partido de las ventajas que brinda Internet, evolucionar pasando de opciones limitadas a nichos de mercado. La Wiki es explícitamente una de las herramientas de integración más recomendadas para la aplicación de este patrón.

8- Modelos livianos y escalabilidad rentable.
La Wiki es una solución que aplica a este patrón ya que con 'casi nada' se logra mucho. Se crean contenidos, se genera conocimiento colectivo y se fomenta el efecto de red.

Desde el punto de vista de la infraestructura necesaria para montar el sistema de Wiki se debió instalar un servidor acorde a los requisitos exigidos por la herramienta. Para esto se armó un ambiente LAMP [24] que permite brindar el soporte necesario. Esta combinación de tecnologías (Linux como sistema operativo, Apache como servidor Web, MySQL como servidor de base de datos y PHP como lenguaje de programación) fueron instaladas en una máquina virtual que se dejó operativa en la red de la empresa.

4.2 Ajustes

Las funcionalidades de la Wiki han sido restringidas y ajustadas a los requerimientos solicitados por *EviMed.* No se trata de una Wiki abierta a modificaciones por parte de cualquier usuario, como es común en la mayoría de las Wikis. Es una Wiki restringida en la que solo se le permite realizar modificaciones a un grupo concreto de usuarios, que varía de acuerdo a los cursos que se desarrollan y los médicos que se inscriben a ellos.

4.2.1 Permisos

Cualquier persona puede acceder a la Wiki con permiso de sólo lectura. Existe un grupo de usuarios definidos en la base de datos con permisos de lectura/escritura. Un administrador es el encargado de crear los artículos de discusión y los usuarios que se conectan a la Wiki pueden modificar la información editando los artículos existentes, pero no les es posible iniciar nuevos artículos.

No es posible registrarse como nuevo usuario en forma automática. Es desde *EviMed*, a través de un usuario con permisos de administrador, que se pueden agregar nuevos usuarios al sistema.

4.2.2 Inicio

WikiMed cuenta con una página de inicio que es mantenida únicamente por el administrador. En la misma se ofrece al usuario editor ayuda detallada sobre como usar la herramienta.



Figura A4.1: Página de inicio.

Al seleccionar el link 'Registrarse/Entrar' se deberá ingresar nombre de usuario y contraseña. De esa forma, si el usuario tiene los permisos adecuados, podrá navegar entre los distintos artículos y editar aquellos en los que desee aportar.

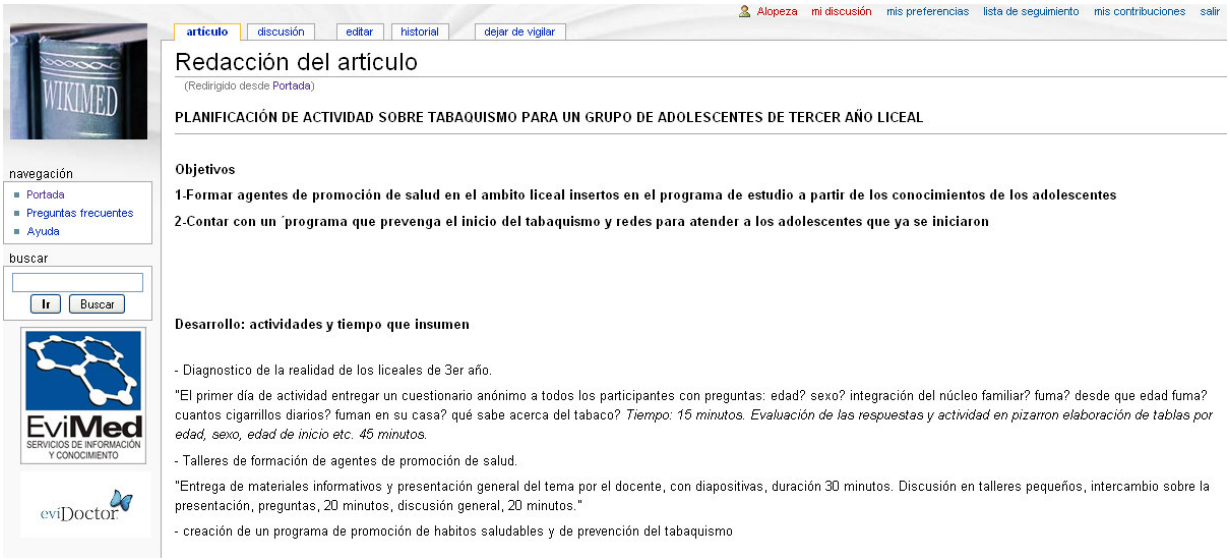


Figura A4.2: Portada de la Wiki.

El usuario administrador podrá editar la portada, seleccionando la pestaña 'Editar', disponible en la interfaz. Esta pestaña no es visible al resto de los usuarios del sistema.

4.2.3 Artículos

Las *consignas* propuestas por *EviMed* son artículos que pueden ser modificados por los usuarios editores. Además de discutir estos artículos, es posible ver su historial de modificaciones, comparar diferentes versiones de los mismos, o incluso identificar quién hizo qué modificaciones. También se cuenta con la funcionalidad de vigilar las modificaciones del artículo, recibiendo un mail cada vez que este es modificado por alguien.

Las siguientes figuras muestran un ejemplo de la dinámica seguida en la edición de determinado artículo.

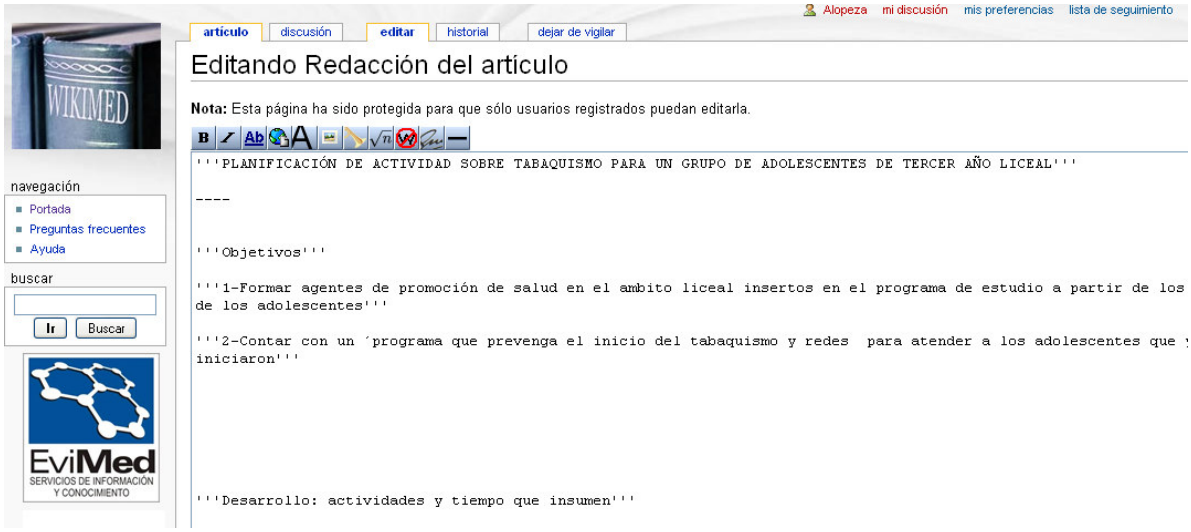


Figura A4.3: Editar redacción de un artículo.

Para que el artículo efectivamente quede modificado se deberá grabar la página.

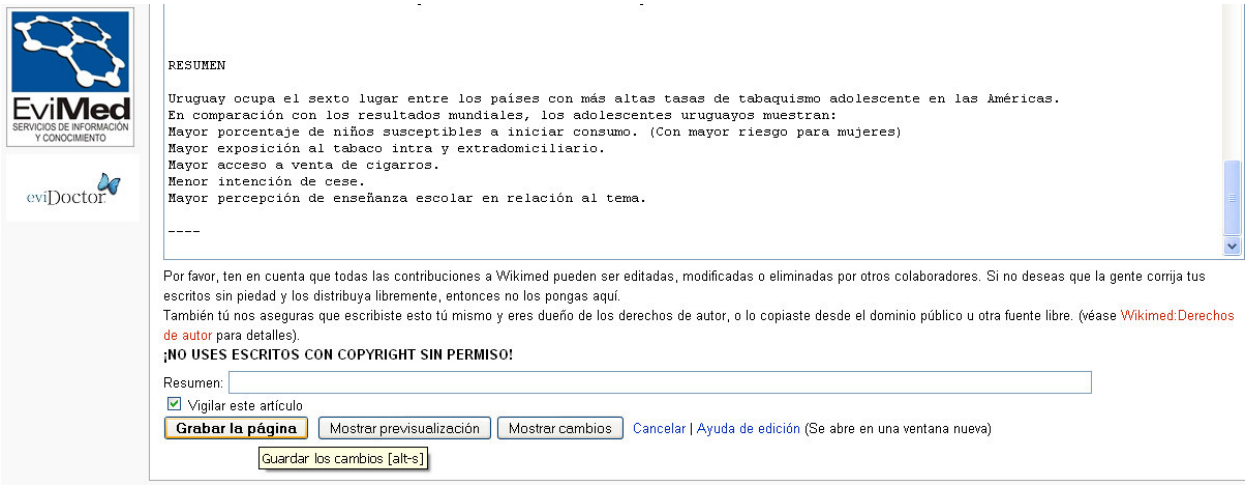


Figura A4.4: Grabar página.

Para discutir y aportar en el artículo es necesario ingresar a la pestaña discusión, y seleccionar el cuadro de texto '+'.

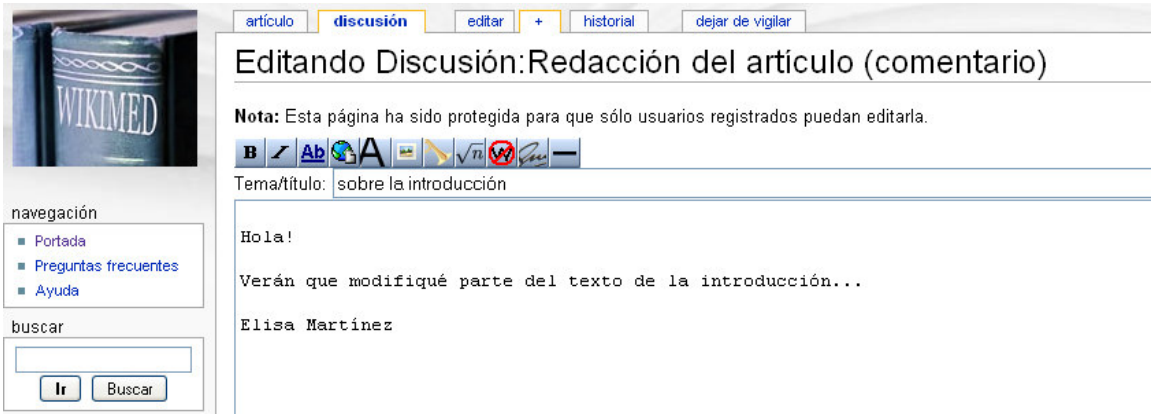


Figura A4.5: Editar discusión de un artículo.

Una vez grabada la página se puede ver el artículo original con las modificaciones realizadas.

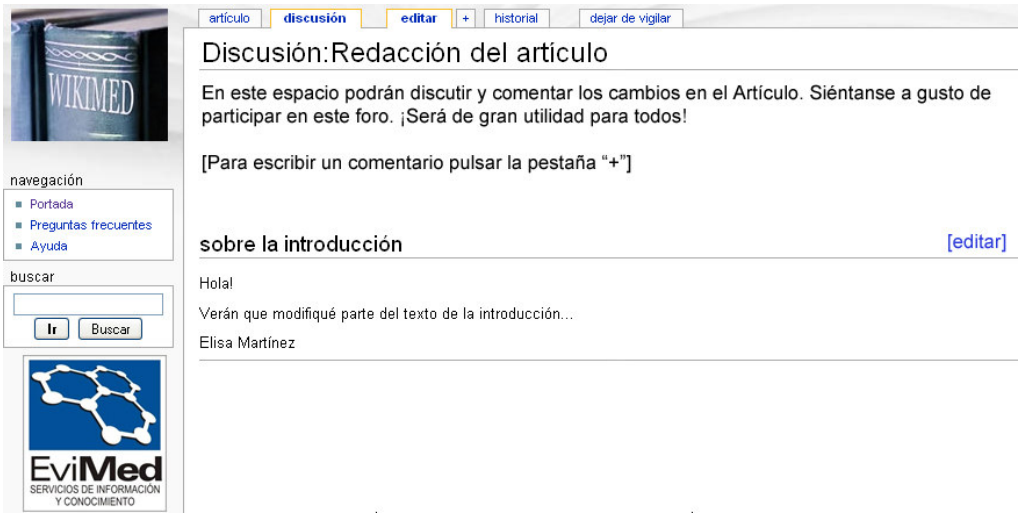


Figura A4.6: Artículo editado.

De esta manera, los usuarios de *EviMed* pueden mediante *WikiMed* participar en la creación de artículos en forma colaborativa. Al agregar contenido relevante y fidedigno relacionado con los temas expuestos en dichos artículos, se genera conocimiento colectivo.

4.3 Decisiones tomadas

Durante el transcurso del presente trabajo, se presentaron ciertas dificultades que debieron ser sorteadas tomando decisiones que permitieran avanzar hacia los objetivos trazados.

En el planteo inicial se decidió investigar la posibilidad de usar Genexus Rocha para el desarrollo de un sistema que apuntara hacia las nuevas tendencias Web 2.0 por tratarse de una solución nacional, aportando de esta forma, un valor agregado al resultado obtenido en el trabajo. A partir de un análisis exhaustivo de la herramienta, esta no fue considerada dado que al momento del presente estudio se encontraba en versión no estable y con poca documentación disponible [ver anexo A1-3. Funcionalidades Web 2.0 de Genexus].

Con el propósito de mejorar la forma en que los usuarios de *EviMed* actualmente acceden, descargan y comparten la información dispuesta en distintos formatos de archivos, se incursionó en la evaluación de iTunes University [ver anexo A1-4. Estudio de iTunes U]. Esta alternativa fue desestimada por requerir, en el momento de su estudio, la formalización de un acuerdo con Apple para la creación del sitio.

Para finalizar esta etapa y con el resultado de una revisión completa del estado del arte, se optó por abandonar la integración del producto *Moodle* con *MediaWiki*, dejando de lado la posibilidad de desarrollar el módulo de autenticación única. La existencia de una alternativa ya implementada y de acceso libre que resuelve esta funcionalidad motiva a postergar esta tarea de integración para cuando la empresa lo considere conveniente, quedando así, planteada como trabajo a futuro.

5. Pruebas y resultados

Una vez configurada la personalización de la Wiki por medio de los ajustes antes descritos, se procedió a la realización de un conjunto de pruebas básicas con el fin de verificar el comportamiento deseado. Para concluir la sección se detallan los resultados obtenidos luego de la puesta en producción y utilización de *WikiMed*.

5.1 Pruebas

Se detallan a continuación las pruebas realizadas previamente a la puesta en producción de la Wiki y los resultados obtenidos en el uso de *WikiMed* por los usuarios de la red social *eviDoctor*.

Una vez investigada la herramienta *MediaWiki* de la fundación *WikiMedia* y establecidas las funcionalidades disponibles, se realizó la configuración acorde a las características de la población que iba a utilizarla. Basados en estas características se elaboraron un conjunto de casos de uso para probar el funcionamiento de la Wiki.

El testeo fue realizado y dividido según los roles definidos: administrador y editor. Las funcionalidades disponibles en la Wiki dependen del perfil de usuario registrado. Inicialmente se crearon dos usuarios, uno con cada rol y se realizaron las pruebas correspondientes. También se utilizó el concepto de usuario no registrado, cuyo testeo consiste en intentar usar las funcionalidades que provee la Wiki sin registrarse en ella.

Para el usuario con rol administrador, se verificó que pudiera:

- Dar de alta, modificar y dar de baja usuarios.
- Editar la portada de la Wiki.
- Asignar o quitar permisos a los usuarios.
- Crear nuevos artículos o páginas.
- Personalizar la interfaz del menú principal, tanto en los links a sus páginas, como en logos que identifican a la empresa *EviMed*.

En lo que respecta al usuario con rol editor se verificó que tuviera la posibilidad de editar una página o artículo ya creado por el usuario administrador.

Además de verificarse que no pudiera:

- Registrar nuevos usuarios.
- Crear nuevos artículos o páginas.
- Editar la portada.

Finalmente se realizaron pruebas accediendo a la Wiki sin registrarse como usuario. Se constató que únicamente fuera posible acceder a la lectura de artículos.

Otras pruebas generales realizadas y que no se relacionan con roles creados fueron:

- Que los usuarios existentes en *EviMed,* migrados desde la base de la empresa pudieran ingresar a la Wiki.
- Que los nuevos usuarios registrados a través de la herramienta lo hicieran de igual forma.
- Que la ubicación de los logos de *eviDoctor* y *EviMed*, que caracterizan la empresa, fuera la apropiada.
- Se testearon otras funcionalidades complementarias.

5.2 Resultados

A través del usuario administrador se exploraron las funcionalidades que se describen en la documentación de la herramienta *MediaWiki* con el propósito de adquirir conocimiento

sobre ellas. De esta manera fue posible guiar sobre el uso de la Wiki al personal de *EviMed* para la generación de 'ayudas', que fueron incluidas en la misma Wiki. La empresa elaboró un video en donde se explica el funcionamiento de la Wiki. En la siguiente imagen se puede ver una captura del mismo.

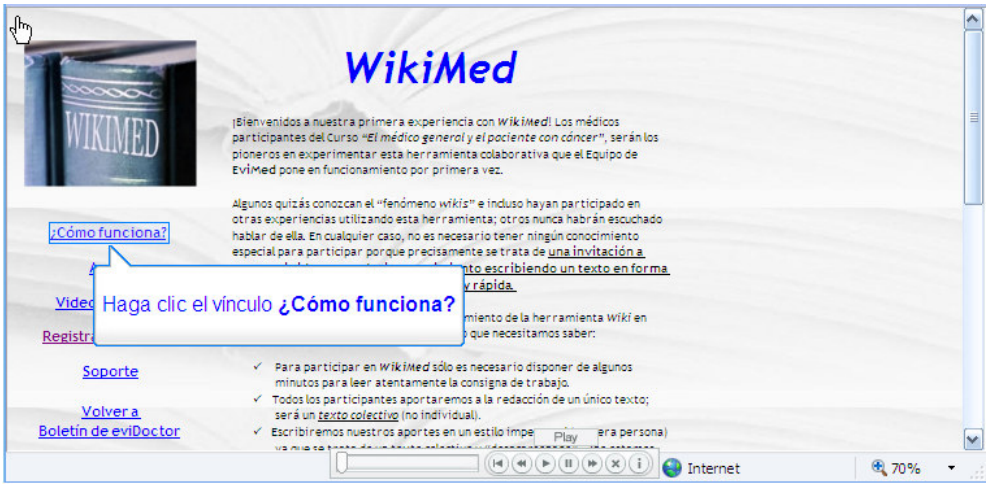


Figura A5.1: Ayuda en WikiMed

Los resultados de la puesta en producción de *WikiMed* no se hicieron esperar. A pocos días de su puesta a punto fue utilizada como apoyo educativo en el marco del curso 'El médico general y el paciente con cáncer' organizado por *EviMed* y la Comisión Honoraria de Lucha Contra el Cáncer. Este curso se realizó entre junio y principios de agosto de 2008, siendo la primera experiencia de los médicos de la red social *eviDoctor* en *WikiMed*.

La propuesta de los organizadores fue la creación de un artículo construido en forma grupal y que tuviera como consigna: 'Estrategias para mejorar la comunicación entre los integrantes del equipo de salud, ante el paciente con cáncer'. El artículo escrito contó con la participación de nueve médicos ubicados en distintos departamentos del país.

La segunda experiencia con *WikiMed* se realizó del 3 de octubre al 17 de noviembre de 2008 en el marco del curso 'Cesación de tabaquismo'. La propuesta para el uso de la herramienta fue la misma que en el curso anterior y la nueva consigna fue 'Cómo organizar una Unidad de Tabaquismo'.

En este caso se contó con 186 inscriptos al curso, 19 de los cuales fueron participantes activos en la Wiki, dos editoras y 3.172 palabras generadas en la creación final del artículo. Dicho artículo tuvo la participación de médicos radicados en ocho departamentos del Uruguay.

El contenido de los artículos, los comentarios que los propios usuarios registrados agregaron, así como los datos estadísticos sobre ambas experiencias se presentan en forma de anexo [ver anexo A5-1. Experiencia en WikiMed]. Es posible además ver la aceptación por parte de los médicos y el uso de la Wiki como forma de interactuar.

El uso de la Wiki se fija como una nueva actividad dentro de cursos realizados por la empresa, donde se propone a los usuarios crear un nuevo artículo en base a una consigna, generando un trabajo totalmente nuevo. Además, en estos artículos existen revisiones hechas por el coordinador del curso, quien es experto en la materia, verificando que los conocimientos plasmados en el artículo sean correctos. Esto asegura que el contenido no tenga errores conceptuales. En base a las experiencias realizadas se puede notar que los cambios registrados en estas revisiones son menores, lo que en cierta medida, es comparable al resultado publicado por Giles.

Parte B - CÁMARA DE GESELL



2. Desarrollo de la investigación

2.1 Marco teórico

El intenso contacto con los niños, desde su lugar como padre, maestro, psicólogo y pediatra, despertó en Arnold Lucius Gesell [25] un gran interés por el desarrollo de los mismos. En su afán por conocer el comportamiento de los niños sin la interacción de terceras personas (ya sea padres, médicos, maestros u otros), el doctor Gesell (1880 – 1961) puso al servicio de su investigación la más sofisticada tecnología a su alcance. Utilizó los últimos avances de video y fotografía de su época como parte de lo que llamó Cámara de Gesell. Esta cámara consistía en dos ambientes contiguos y separados por un vidrio de visión unilateral de un salón hacia el otro, para promover y facilitar un desarrollo más natural de la actividad observada.



Figura B2.1: Ejemplo Cámara de Gesell.

Con el paso del tiempo, se fueron incorporando otros usos a la cámara, pero manteniendo su objetivo principal intacto; realizar observaciones en vivo de personas interactuando en situaciones reales. Además de seguir siendo un método vigente en el área de la psicología infantil, esta invención de principios del siglo XX, ha servido como herramienta en la resolución de casos a nivel judicial. En muchos países la implementación de la Cámara de Gesell ha significado un gran avance en los procesos de investigación, que se traducen en un paso adelante en la protección de personas víctimas de delitos [26]. Otro de los usos más frecuentes de este dispositivo es en la investigación cualitativa en ciencias sociales, lo que se conoce como grupos focales, ampliamente conocidos en el área de las investigaciones de mercado.

La Asociación de Psiquiatras del Interior (API) planteó la dificultad de realizar instancias clínicas en las que participen varios profesionales. Por la naturaleza de esta asociación, sus integrantes están dispersos por el país, dificultando la realización de interconsultas, ateneos clínicos y otras actividades que requieran presencia física de todas las partes. Bajo esta consigna se plantea como proyecto de grado poder proveer una herramienta informática que simule una Cámara de Gesell a través de Internet, que resuelva además los problemas particulares de este grupo de usuarios. La idea de los psiquiatras detrás de esta inquietud se traduce en que un grupo de usuarios pueda observar una consulta virtual a través de Internet proporcionada por otro usuario, con el requerimiento mínimo de un bajo ancho de banda pero que a su vez permita obtener una resolución aceptable en las imágenes de video. Esta idea se potencializa incorporando otras posibilidades como ser: la interacción con el médico

entrevistador durante el desarrollo de la sesión, mediante intercambio de mensajes; la posibilidad de descargar localmente las emisiones para su posterior reproducción (de forma desconectada) y que puedan ser consumidas con una buena calidad de video sincronizado con los comentarios generados durante la entrevista en línea; interfaces de usuario amigables y buena usabilidad, necesarias para que cualquier tipo de usuario del sistema pueda crear la emisión de su propia entrevista, contando simplemente con una conexión a Internet, una cámara web y un micrófono; la asociación de un conjunto de datos a la emisión de la entrevista que permita a los usuarios tener información extra sobre el tema que se está abordando; la posibilidad de definir los datos asociados tanto en forma conectada como desconectada; la potencia de búsquedas sobre la información asociada a cada entrevista de manera de contar con un repositorio que pueda ser usado como fuente de conocimiento; una definición de seguridad que permita mantener la confidencialidad de datos sensibles del conocimiento público, como pueden ser las historias clínicas de los pacientes; la posibilidad de compartir publicaciones de interés propias o de terceros con otros médicos. Estas funcionalidades son un claro ejemplo del valor agregado que se incorpora a la idea original.

El desarrollo de este proyecto puede proveer beneficios también desde el punto de vista educativo. Dado que la mayor parte de las actividades formativas se desarrollan en Montevideo, esta es una forma de favorecer el intercambio en la Educación Médica Continua y de enriquecer los conocimientos facilitando una comunicación más fluida. Además permitirá generar una fuente de referencia importante y siempre disponible. Los datos del sistema serán valiosos pues tienen implícito el valor de la información obtenida de profesionales de la salud.

2.2 Estado del arte

Se comenzó estudiando las funcionalidades que proveen algunas herramientas de mensajerías comunes como ser Messenger [27] y Google Talk [28]. Estas básicamente brindan comunicación a través de chat y videoconferencias, permitiendo que se conecten varios usuarios en una misma sesión.

Teniendo en mente el conjunto de funcionalidades planteadas se amplia la investigación a una serie de herramientas y servicios que se ofrecen actualmente en el mercado buscando alguno que provea más que chat y videoconferencia. A continuación se mencionan algunas de las herramientas que se investigaron y los detalles relevantes de las mismas.

ooVoo [29] es un software de conferencia web disponible para sistemas operativos Windows y Mac. Permite la comunicación mediante chat, video y audioconferencia donde las sesiones de video pueden ser grabadas y compartidas con otros usuarios. También se pueden enviar archivos de gran tamaño entre usuarios. No cuenta con límite de contactos y permite añadir seguridad para indicar quien puede contactar a un usuario. Se cuenta con cuatro planes de uso, uno gratuito y los otros tres con costos y funcionalidades crecientes. Se detallan en la siguiente tabla las funcionalidades incluidas en cada plan.

	Plan 1	**Plan 2**	**Plan 3**	**Plan 4**
Número de streaming (usuarios)	2	3	4	6
Largo de mensajes de video (minutos)	1	5	5	5
Tamaño de almacenamiento de video (MB)	0	500	750	1000
Tamaño de transferencia de archivos (MB)	0	10	20	25
Costo (dólares mensuales)	Sin costo	8	13	18

Acrobat.com [30], también denominada suite ofimática o suite de oficina, fue lanzado en junio de 2008 por la firma estadounidense Adobe Systems. Es una herramienta colaborativa que reúne un conjunto de servicios en línea con el que se puede crear y compartir documentos, simplificando el trabajo con ayuda de otros usuarios. Este conjunto de servicios incluye:

- Share: permite enviar archivos a otros usuarios facilitando el acceso pero manteniendo el control sobre sus contenidos.
- Adobe Buzzword: permite escribir y colaborar en cualquier tipo de documentos, en cualquier momento y lugar.

- My Files: permite almacenar archivos en línea.
- Create PDF: sirve para convertir archivos al formato PDF en línea.
- Adobe ConnectNow: uso compartido de pantalla, chat, notas, audio y video para llevar a cabo reuniones en línea.

En julio de 2008 se lanzó Brio, es la nueva generación de Adobe Acrobat ConnectNow. Es un sistema para mantener una reunión entre varias personas cada una situada en lugares diferentes. Es una versión Beta que, a través de una URL, permite reuniones de hasta tres personas compartiendo pantalla, chat y pizarra. El sistema integra VoIP (voz sobre IP), teleconferencias y video multi-punto (un solo transmisor y dos o más receptores). Se trata de una habitación virtual que alberga a personas ubicadas en distintos lugares. Este servicio dispone de tres tipos de versiones.

- Versión gratuita: sólo dos personas pueden trabajar en línea pero no pueden crear archivos PDF.
- Versión básica Premium: cuesta 15 dólares mensuales, permite reuniones web con hasta cinco participantes y conversiones en línea de 10 documentos a PDF al mes.
- Versión Premium Plus: cuesta 39 dólares al mes o 390 dólares al año, tiene capacidad para reuniones de hasta 20 personas y creación en línea ilimitada de archivos PDF.

VSee [31] de la Universidad de Stanford, es una aplicación gratuita de videoconferencia y un servicio para compartir aplicaciones que permite la comunicación con hasta seis personas. Se puede editar y anotar documentos en forma remota, compartir aplicaciones y escritorios enteros, transferir archivos, grabar y compartir videos. El chat por texto puede realizarse entre todos o uno a uno. En el manejo de cámaras remotas, se puede mostrar una vista panorámica, inclinar o incluso hacer zoom. El sistema corre sobre plataforma Windows y se conecta a un servicio a través de Internet. La comunicación entre usuarios se realiza vía P2P.

La compañía SightSpeed Inc. cuenta con el producto SightSpeed [32], un sistema de videoconferencia multiplataforma. Este producto permite realizar llamadas de video con hasta cuatro personas, chatear, compartir archivos, grabar sesiones y enviar mensajes de video. Es gratis para dos personas y cuesta 10 dólares por mes para las cuatro, permitiendo almacenamiento ilimitado de video.

PalBee [33] provee un sistema de videoconferencia en línea, permite crear sesiones de video con hasta 10 personas, en la que todos pueden usar la pizarra, cargar presentaciones de PowerPoint, grabar durante una hora todo sucedido en la conferencia y publicarlo luego en YouTube. El usuario que quiere compartir una sesión con otras personas puede seleccionarlas desde una lista de contactos asociados. El servicio es totalmente gratuito por el momento pero aún se encuentra en versión Beta.

Existen varias herramientas más de videoconferencia y chat que proveen funcionalidades similares a las antes descritas. A través de un usuario registrado es posible obtener una URL correspondiente a la sala en la que se llevará a cabo la conferencia y donde los usuarios invitados deben ingresar para presenciarla. Estas herramientas son aplicaciones Web y algunas de ellas de uso libre. MeBeam [34] soporta 16 usuarios. Vidivic [35], aún en versión Beta, permite reuniones de nueve participantes, pero soporta reuniones de hasta cuatro horas. MegaMeeting [36] permite videoconferencias de hasta 16 personas en simultáneo, pero un número ilimitado usuarios pueden visualizarla. Además, esta herramienta no es gratuita como las dos anteriores, sino que cuenta con una gran variedad de planes dependiendo de la cantidad de usuarios y la forma de uso del servicio.

Skype [37] es un servicio VoIP y videoconferencia que se puede utilizar para tener llamadas de video uno a uno. Aunque las videoconferencias están limitadas para dos personas, las audioconferencias soportan hasta cinco usuarios. También permite la transferencia de archivos. A principios del 2009 se realiza una actualización que mejora notoriamente la calidad de imagen de video y llamadas de voz, si la velocidad de la conexión es suficiente. Se incorpora también la funcionalidad de '*Picture in* P*icture*' donde se pueden ver la imagen de la

persona contactada y la de sí mismo. La limitación de videollamadas limitadas a dos personas corresponde a la infraestructura P2P, pero actualmente es una prioridad a solucionar en futuras versiones. Es de uso libre y para siguientes versiones la compañía responsable promete llegar a diferentes dispositivos, así como también la funcionalidad de grabar sesiones [38].

SnapYap [39] cuenta con menores funcionalidades comparadas con Skype. Es una herramienta web que aunque permite llamadas con audio y video uno a uno, no cuenta con mensajes de texto ni transferencias de archivo.

A continuación se muestra un cuadro comparativo de las características fundamentales para las herramientas mencionadas teniendo en cuenta las siguientes propiedades:

- Número de streaming: especifica el número de cámaras capaces de hacer streaming en simultáneo. En caso de que existan varias versiones se tiene en cuenta la cantidad máxima de usuarios que soporta.
- Chat: indica si integra una función para que los usuarios envíen mensajes de texto escritos a los demás participantes.
- Transferencia de archivos: indica si se permite enviar archivos a otras personas durante la reunión.
- Grabación de la sesión: indica si permite grabar todos los canales de audio y video.
- Basado en Web: determina si necesita descargar un software cliente para poder utilizar el servicio o si se ejecuta directamente sobre Internet a través de los navegadores.
- Costo: indica el costo máximo en caso de que haya más de una versión disponible.
- Versión Beta: indica si la herramienta se encuentra aún en versión Beta.

	Streaming	Chat	Transferencia de archivos	Grabación de sesión
Oovoo	6	Si	Si	Si
Acrobat.com	20	Si	Si	No
VSee	6	Si	Si	Si
SightSpeed	4	Si	Si	Si
PalBee	10	Si	Si	Si
MeBeam	16	Si	No	No
Vidivic	9	SI	No	No
MegaMeeting	16	Si	No	No
Skype	2	Si	Si	No
SnapYap	2	No	No	No

	Basado en la Web	Costo	Versión Beta
Oovoo	No	18 dólares mensuales	No
Acrobat.com	Si	39 dólares mensuales	No
VSee	No	Sin costo	No
SightSpeed	No	10 dólares mensuales	No
PalBee	Si	Sin costo	Si
MeBeam	Si	Sin costo	No
Vidivic	Si	Sin costo	Si
MegaMeeting	Si	80 dólares mensuales	No
Skype	No	Sin costo	No
SnapYap	Si	Sin costo	No

[40]

También se investiga la herramienta Vuze [41] como ejemplo de aplicaciones que permiten descarga de contenidos. Este programa permite intercambiar videos y que los usuarios puedan agregarles comentarios, asignarle categorías y calificarlos. Se trata de un cliente de descarga BitTorrent, lo que significa que a través de 'redes de pares' los usuarios pueden compartir sus propios archivos con otros usuarios sin subirlos a ningún sitio específico. Es preciso que el cliente instale el componente Máquina Virtual Java.

Si bien cada una de las aplicaciones investigadas resuelve algunas de las necesidades planteadas originalmente, ninguna de ellas puede modelar la idea en su totalidad. En un mismo software no se encuentran resueltas la transmisión en vivo de uno a muchos, la posibilidad de descargar contenido multimedia para poder ser consumido desconectado o mantener información asociada a dicho contenido. Tampoco integran la posibilidad de definir grupos cerrados de usuarios, ni la de compartir información genérica además de las entrevistas como pueden ser las publicaciones.

Buscando otras alternativas se llega a una herramienta llamada Elluminate [42] que ofrece colaboración entre diferentes usuarios y brinda funcionalidades dentro de la industria del e-learning (aprendizaje electrónico). Esta herramienta permite la posibilidad de crear salas virtuales, que pueden ser compartidas con otros usuarios, donde se cuenta con teleconferencia que incluye video, audio y chat. Una vez que se está presente en la sala, es posible enviarle mensajes al resto de los usuarios participantes o incluso a alguno en particular. En estas sesiones es posible compartir aplicaciones, archivos de video, multimedia y pizarras electrónicas. Los usuarios pueden utilizar un recurso al mismo tiempo. Es posible grabar la actividad generada en las videoconferencias y posteriormente publicarla. Soporta varios formatos de audio, video y podcast, que al quedar disponibles, los usuarios pueden descargarlas para ser vistos en PC o iPod. Existen dos versiones de esta herramienta: la versión gratuita, que puede ser usada por tres personas y la versión Premium, donde su costo aumenta de acuerdo a la cantidad de participantes.

Continuando con una investigación más minuciosa guiada por la tendencia del proyecto, se llega a una empresa llamada Polycom [43] que ofrece soluciones de "Telepresencia". Esta solución permite realizar conferencias entre varias personas que se encuentran en diferentes lugares, como si estuvieran en una misma habitación, Polycom cuenta con:

- Polycom RPX HD Serie 400: es ofrecido a ejecutivos, reuniones de juntas, formación, educación, gestión de proyectos. Es la primera y única solución de telepresencia inmersiva (control de todo el ambiente en ambos lados de la pantalla). Los participantes pueden estar sentados, de pie o caminar por la habitación. Permite a los participantes ver las expresiones faciales, contacto visual y leer el lenguaje corporal de los otros. Cuenta con pantalla gigante. Su capacidad es para 8 a 28 participantes.
- Polycom RPX HD Serie 200: Misma experiencia que la serie 400 pero en un entorno más pequeño ideal para pequeñas y medianas organizaciones y empresas con los grupos de trabajo dispersos. Pantalla de video de menor tamaño que la serie 400. Su capacidad es para 4 a 18 participantes.
- Polycom Experiencia Telepresencia TPX 306: ideal para pequeñas reuniones de grupos dentro de grupos de trabajo dispersos geográficamente. Pantalla plana de 60 pulgadas crea la ilusión de estar en la misma habitación. Su capacidad es de hasta seis participantes.

Estas dos últimas aplicaciones analizadas son las que más se aproximan al enfoque planteado en el proyecto. No obstante, cuentan con funcionalidades que no se consideran adecuadas para la resolución del problema.

Los usuarios asociados en API forman una comunidad, un grupo cerrado de usuarios con intereses y una profesión en común. Elluminate no maneja el concepto de comunidad cerrada, más bien permite que cualquier usuario pueda compartir información con otro, pero no con un concepto de grupo predefinido.

A pesar de que con Elluminate es posible grabar sesiones realizadas y publicarlas posteriormente, no se forma un repositorio de información más allá de las sesiones de

videoconferencia. No se provee una manera de asociar información a las emisiones en vivo sobre las que luego se puedan realizar búsquedas avanzadas basadas en el contenido o en etiquetas asociadas a la información. Además se considera un inconveniente el costo que puede significar contar con un número importante de usuarios, dado que como fin se busca motivar la participación del mayor número posible de estos de manera de poder apuntar a la Educación Médica Continua.

Como desventajas principales en el producto Polycom se destaca que no tiene en cuenta la posibilidad de usar como dispositivo de entrada otro que no sea cámara de video. Además tampoco se maneja la posibilidad de asociar información a las sesiones.

Ante la falta de una aplicación que reúna las características planteadas y pensando además en una solución genérica que pueda ser usada en cualquier ámbito donde sus funcionalidades sean aprovechables, se ve motivado el desarrollo de un nuevo software. La idea de crear una nueva aplicación que desarrolle el concepto de *Cámara de Gesell* virtual. El hecho de llamarlo igual al invento del médico estadounidense va de la mano de su concepción más fundamental, una sesión desarrollada en dos habitaciones separadas -aunque en este caso sea a cientos de kilómetros- y la visión unilateral, puesto que ni el médico tratante, ni su paciente podrán ver la audiencia que los observa (aunque el médico, por supuesto, puede saber quiénes de sus colegas están participando a distancia).

Desde un inicio la solución se piensa para ser usada de manera genérica, de forma que pudiera ser implantada en cualquier disciplina. Es por esto que se considera brindar la posibilidad de definir el significado de la información que se maneja desde el propio sistema, definiendo la metadata de los datos. El proyecto se inicia influenciado por el entorno médico pero se define de manera genérica. Es así que puede ser usado en cualquier entorno donde se pretenda implantar medios de comunicación y creación de conocimiento colectivo.

Dado que el sistema debe recibir y enviar información entre muchos usuarios, y que la información debería estar siempre disponible, se decide resolver el sistema en un entorno cliente-servidor. Teniendo en cuenta que se pretende obtener un producto que brinde una buena usabilidad e interfaces amigables para el usuario se opto por construir un sistema que se asemejara a una aplicación de escritorio. Este sistema también deberá poder contar con algunas funcionalidades cuando se trabaje desconectado. Es de esta manera que se plantea la creación de un 'Smart Client' (cliente inteligente) [44].

G-Sell plantea resolver los problemas que padecen hoy en día los médicos debido a situaciones geográficas (grandes distancias que impiden mayor frecuencia en sus encuentros presenciales), problemas organizativos, costos de traslado y disponibilidad de los participantes, entre otros. También facilitar el consumo de contenido multimedia generado por los propios usuarios sin necesidad de estar conectados a Internet. Respecto a esto último se apunta a solucionar la problemática actual de médicos que residen en el interior del país, especialmente los que se encuentran en ciudades pequeñas, o en zonas rurales donde aún no ha llegado la posibilidad de contar con conexión a Internet o bien la misma es lenta o cara. Los médicos se trasladan esporádicamente a la ciudad. Es entonces cuando podrían sincronizar la información local con la disponible para ellos en el servidor, luego estarían en condiciones de consumir la información localmente. Asimismo, este problema del acceso a conexión de Internet es complejo para los médicos en general, aún los que residen en ciudades, ya que tienen jornadas de trabajo muy extensas, traslados entre consultorios donde no siempre tienen disponible equipos con conexión.

Ya con el alcance del proyecto establecido, con el estado del arte enfocado a solucionar las necesidades planteadas por la API; se continuó la investigación principalmente hacia el entorno de herramientas de desarrollo.

Se comenzó con la investigación de los productos Adobe Flex [45] y Adobe Flash [46]. Se buscaba poder crear interfaces RIA, Rich Internet Application (aplicaciones de Internet enriquecidas), punto importante en la aplicación a crear, dado que se trata de usuarios que deben encontrar una buena usabilidad del sistema.

Flex permite crear dichas interfaces y Flash animarlas. La investigación respecto a ambos productos, a grandes rasgos, mostró que aportaban al desarrollo de *G-Sell* sobre todo

pensando en la creación de interfaces RIA. Es así que se decidió continuar con la investigación dentro de Adobe, convergiendo a una nueva tecnología denominada Adobe AIR (Adobe Integrated Runtime) [47]. Su tendencia se enfoca hacia un modelo mixto donde puedan coexistir aplicaciones de escritorio con aplicaciones web. Aprovecha las mismas tecnologías que se usan en la red (Flash, Flex y Ajax) para desarrollar aplicaciones de escritorio. Esto significa que 'aparentemente' reunía las características principales para el desarrollo de *G-Sell.* Sin embargo, su versión final había sido lanzada a principios de 2008 siendo una herramienta de desarrollo muy joven en el mercado, motivo que determinó su descarte y la continuación de la investigación.

Se contactó y visitó la empresa nacional de desarrollo de software d2B Network [48], quienes por su experiencia en creación de interfaces RIA y aplicaciones multimedia brindaron su opinión acerca del desarrollo con herramientas Microsoft, del lenguaje C sharp, del uso de Windows Presentation Foundation (WPF) para aplicaciones RIA y la creación de 'Smart Client' a través de .NET.

Sin duda existen otras alternativas de desarrollo que se adecuan al proyecto, como puede ser el entorno java, pero se decidió detener la investigación sobre herramientas y dedicar el tiempo restante a aprender sobre las tecnologías Microsoft. En esta decisión influyó la gran repercusión de las tecnologías lanzadas por Microsoft en el mercado. Si bien se conocía el framework de .NET, no se tenía experiencia en el lenguaje C sharp, o incluso en el uso de herramientas más específicas como ser LINQ y Windows Presentation Foundation.

Fue así que se llegó a Visual Studio 2008, una herramienta que provee el entorno apropiado necesario para el desarrollo de *G-Sell.* Se detallan a continuación las funcionalidades a utilizar:

- WCF: para la creación de los servicios Web [ver anexo A2-4. Introducción a Windows Communication Foundation (WCF)].
- WPF: para la construcción de interfaces de usuarios más ricas y dinámicas que las conocidas hasta el momento, facilitando la usabilidad de la aplicación [ver anexo A2-3. Windows Presentation Foundation (WPF)].
- LINQ: herramienta que permite generar código en forma automática a partir de modelos que se definen tomando como base, orígenes de datos ya creados.
- Integración con gestores de base de datos de Sql Server.

Además cabe señalar que se descartó el uso del sistema operativo Linux ya que a los efectos del proyecto no era relevante. Además se tomo en consideración la situación uruguaya en donde la mayoría de los médicos usan sistemas operativos de Windows.

3. Alcance

Definidos los requerimientos inspirados en las necesidades de la Asociación de Psiquiatras del Interior se comenzó con la creación de una 'Cámara de Gesell' virtual. A pesar de que su origen estaba relacionado a grupos de afinidad dentro de la medicina, desde el principio se diseñó esta aplicación pensando en construir una solución genérica que pudiera aplicarse a múltiples ámbitos y realidades. Es por esto que el propio sistema brinda las herramientas necesarias para que los usuarios puedan definir el significado de los datos a ser asociados a las emisiones de video. Un usuario o grupo de usuarios responsable de definir la información necesaria en cada especialidad podrá establecer un conjunto de datos a ser incluidos como información de portada y cierre de las entrevistas.

Las emisiones de video y audio consisten en una persona emitiendo un video hacia un grupo de usuarios que pretenden verlo en línea. Aplicado a la API, la solución construida separa a los participantes de la emisión de una entrevista en un médico con rol 'emisor' y varios médicos con rol 'consumidor'.

El médico 'emisor' cuenta con las funcionalidades de:

- Iniciar una portada con datos asociados a la entrevista que brinden detalles sobre el paciente a evaluar y el tema a ser tratado. La información que aquí se incluya estará atada a la definición de metadata que acompaña a la especialidad. Debe recordarse que los metadatos son definidos por usuarios responsables de manejar el significado del contenido de la información.
- Agendar las emisiones de las entrevistas definiendo día y hora en que van a ser transmitidas y quienes son los colegas invitados a presenciarlas.
- Realizar la emisión en vivo. Una vez finalizada la emisión de video y dado que el servicio de chat funciona independientemente de dicha emisión, el médico 'emisor' y sus colegas podrán mantenerse en línea discutiendo el caso tratado, intercambiando opiniones. Es el médico 'emisor' el responsable de finalizar esta instancia de discusión, finalizando el conjunto de la emisión.
- Finalizar la entrevista agregando información de cierre. Al igual que en la portada, la índole de los datos de cierre están relacionados a la definición de metadata.
- Publicar la entrevista terminada y dejarla disponible para ser consumida a demanda por sus colegas.

Por otro lado, los médicos 'consumidores' tienen disponibles las siguientes funcionalidades:

- Podrán acceder a la emisión en vivo a través de la aplicación en la fecha y horario fijados previamente por el usuario 'emisor'.
- Durante la emisión tendrán la posibilidad de intercambiar información y opiniones con sus colegas usando el servicio de chat. También continuar con el intercambio de mensajes una vez finalizada la emisión de video. Esta funcionalidad estará disponible hasta que el usuario 'emisor' lo disponga.
- Podrán descargar en sus equipos las entrevistas a las que hayan sido invitados y consumirlas localmente para su estudio y análisis.

Además de toda la información referente a una entrevista, existe un área de información donde cualquier usuario puede publicar documentos considerados de interés. Después de proporcionar un resumen sobre el documento podrá subir un archivo en formato Word o PDF. De esta forma los usuarios pueden compartir conocimiento e información de utilidad en sus actividades. Cualquier otro usuario con acceso al sistema podrá ver o descargar este artículo.

De la misma manera se comparten enlaces a sitios o páginas de interés, publicando una descripción del lugar al que se hace referencia y el link correspondiente.

Quedaron fuera del alcance del proyecto la implementación de algunas funcionalidades que se habían pensado incluir en la solución inicialmente. Estas podrán ser desarrolladas en futuras mejoras. Básicamente se trata de las siguientes funcionalidades:

- El reinicio de las publicaciones y descargas de las entrevistas; y la subida y descarga de archivos de interés. Esto facilita la transferencia de información entre los equipos locales y el servidor cuando la misma se ve afectada y no puede ser terminada en forma.
- La funcionalidad de *'Picture in Picture'*, pensada en un inicio para poder visualizar tanto a la persona que está siendo entrevistada como al entrevistador, en caso de contar con más de una cámara.
- La existencia de funcionalidades que permitan moderar los contenidos publicados. Estas funcionalidades permitirían a un usuario administrador eliminar contenidos que hubieran sido publicados equivocadamente, o que no cumplieran con las normas de uso establecidas en la red social. También la posibilidad de que los usuarios denuncien contenidos no válidos o malos usos del sistema.
- La definición de portada y cierre de una entrevista de forma desconectada. Se llegó a resolver las definiciones en forma conectada. También se cuenta con la infraestructura para poder guardar la información en forma local. Lo que queda pendiente es proveer una segunda forma de guardar la información en el caso que no se esté conectado al servidor. Esto es, guardar localmente los datos y luego sincronizarse con el servidor cuando exista una conexión disponible.
- En el momento de agendar las emisiones contar con la posibilidad de generar automáticamente un aviso por correo electrónico a los invitados a participar notificándolos de los detalles de la entrevista. Se decidió dejar la implementación de esta funcionalidad para futuros trabajos de mejora, pero en la interfaz del sistema ya fue tenido en cuenta.
- La actualización de la base de datos con información sobre la cantidad de descargas que han tenido las entrevistas. Esta información ya es tenida en cuenta en las interfaces y en la base de datos. Falta que la información sea actualizada en cada instancia de descarga.
- La posibilidad de emitir un voto en contra o a favor de la entrevista. Ya está representada en la base de datos la posibilidad de llevar un ranking de popularidad. Falta desarrollar la funcionalidad que actualice dicha información.

4. Implementación

Durante el análisis para la implementación de la *Cámara de Gesell* virtual se diferenciaron cuatro módulos independientes que podían ser resueltos por separado y que luego se integrarían para conformar la solución completa.

1- Administración del contenido de la base de datos que se encuentra del lado del servidor (información de usuarios, especialidades, metadata y contenido de la entrevista propiamente dicho).

2- Creación y streaming de multimedia (resolución de la emisión en vivo y almacenamiento paralelo local para posterior emisión a demanda).

3- Creación de salas de Chat (intercambio de mensajes durante la emisión en vivo y almacenamiento de la información para ser usada durante la emisión a demanda).

4- Administración del contenido de la base de datos que se encuentra del lado del cliente (para proveer la información necesaria para el consumo de los datos locales que hubieran sido descargados anteriormente).

Partiendo de este enfoque se procedió en una primera etapa a diseñar la arquitectura completa de la aplicación. La propia naturaleza de la misma (cliente-servidor) determinó la previa instalación, configuración y prueba de cada servidor que ofrece los servicios a las componentes identificadas.

Dado que se desea brindar a los usuarios la posibilidad de consumir contenido tanto cuando el sistema se encuentra conectado como también cuando se encuentre trabajando en forma desconectada, es necesario contar con cierta inteligencia en el extremo cliente. De esta manera se puede aprovechar de mejor forma el rendimiento de un equipo con características aceptables, sin depender del rendimiento que se pueda obtener a través de la red, considerando sobre todo la velocidad de respuesta a las peticiones de los usuarios. La búsqueda de mejores rendimientos es notoria cuando se pretende consumir emisiones de video a demanda. La posibilidad de consumir los videos descargados localmente usando una definición de imagen mucho mejor que la obtenida en la entrevista en vivo ejemplifican este punto. Además se tiene en cuenta la búsqueda de una buena usabilidad tratando de alentar el uso del sistema por parte de los usuarios. Se busca brindar interfaces de usuario ricas. También ocurre que aunque una aplicación Web podría acceder a información en cada equipo cliente, la seguridad que se hubiera configurado en estos equipos podrían no ser suficientes para el acceso a los datos. El volumen de información que administra el sistema también puede llegar a ser considerable, por lo que es beneficioso mantener un modelo relacional similar al que se tiene del lado del servidor, pero del lado del cliente. Estas consideraciones motivan la incorporación de tecnologías para la construcción de una aplicación cliente-servidor del estilo *Smart Client*. De esta forma se aprovecha lo mejor de las aplicaciones de escritorio y lo mejor de las aplicaciones Web.

Una vez culminada esta etapa se diseñaron y validaron las distintas interfaces de usuario, para continuar con el diseño e implementación independiente de cada módulo antes mencionado. En forma paralela se implementaron las interfaces ya validadas con el fin de integrar con ellas cada uno de estos módulos desarrollados y lograr así el objetivo buscado, la creación del sistema *G-Sell*.

4.1 Descripción del entorno de desarrollo

Después de evaluar distintas herramientas para el desarrollo de aplicaciones ricas y alternativas para brindar los servicios que estas consumen, se decidió construir la solución aprovechando las facilidades que brinda el ambiente Visual Studio 2008.

El extremo servidor donde se ejecutan los distintos servicios se decidió implantarlo en una máquina virtual, o sea, un software que permite emular un equipo real. En este caso se uso el paquete de VMware Workstation. Este equipo virtual permite oficiar como nodo servidor del ambiente de producción.

4.1.1 Cliente

El entorno de desarrollo ha de cumplir con los siguientes requisitos:

- Windows XP con Service Pack 2 (SP2) instalado.
- Microsoft .NET Framework v3.5.
- Microsoft SQL Server 2005 Compact Edition.
- DirectX v9 o superior.
- Windows Media Player v9.0 o superior.

Se requiere además las siguientes librerías de terceros desarrolladas para C sharp:

- DirectShowLib v2.0 [ver anexo A4-2. Multimedia en Windows].
- WindowsMediaLib v1.0 [ver anexo A4-2. Multimedia en Windows].
- DotNetZipLib v1.7 [ver anexo A2-5. Publicación y descarga de archivos].

En cuanto a las herramientas de desarrollo, es preciso disponer de la siguiente:

- Microsoft Visual Studio 2008 (desarrollo en C sharp).

4.1.2 Servidor

Se requiere tener instalada la aplicación que administra, gestiona y ejecuta máquinas virtuales en el equipo que aloja la máquina virtual que oficiará de servidor.

- VMware WorkStation v5.0 o superior.

La propia máquina virtual debe contar con el siguiente software como requisito:

- Windows 2003 Server con Service Pack 2 (SP2).
- Microsoft .NET Framework v3.5.
- Windows Media Services9.
- Internet Information Services v6.0.
- SQL Server 2005 Express Edition.
- Microsoft Core XML Services (MSXML) 6.0.
- DotNetZipLib v1.7.

4.2 Arquitectura

Al modelar una solución que representara la realidad planteada se trabajó en la identificación y diseño de una serie de componentes que implementan diferentes funcionalidades bien definidas. Existe un grupo de componentes encargados del streaming de audio y video que permite a cierto usuario del sistema emitir una entrevista mientras que otros, en calidad de invitados, pueden consumirla al mismo momento que esta ocurre.

Una vez que los videos fueron emitidos en vivo, éstos quedan disponibles para ser descargados en forma local y de esta forma consumidos a demanda. Aunque estas funcionalidades están estrechamente relacionadas a las anteriores, son incorporadas por otro conjunto de componentes.

A estas funcionalidades debe agregarse la posibilidad de enviar mensajes vía chat entre los usuarios que están compartiendo la emisión en vivo, además de que dichos mensajes intercambiados estén asociados a la entrevista cuando esta se consume a demanda sincronizados con la visualización de la misma.

Adicionalmente, deben incorporarse algunos componentes que permitan el acceso a datos desde y hacia el servidor, tales como el acceso a datos locales del sistema, en el extremo cliente.

Por último, a través de la capa de presentación o interfaz de usuario del sistema se logra que todos estos componentes sean visibles al usuario final.

De esta forma, los componentes antes enumerados pueden ser agrupados según la función que resuelven:

- Emisión/consumo de streaming de audio y video en línea.
- Consumo de audio y video a demanda.
- Chat durante la emisión en línea.
- Reproducción de chat en video a demanda.
- Acceso a datos del servidor.
- Acceso a datos locales.
- Interfaz de usuario.

También pueden clasificarse los distintos componentes según la interacción que cada uno tenga con los demás y dividirlos en acoplados, débilmente acoplados o desacoplados.

En tal sentido, algunos de estos componentes interactúan entre sí a través de servicios Web, por lo que se dice que tienen un acoplamiento bajo o están débilmente acoplados. Tal es el caso de la interfaz de usuario, la cual cuando accede y presenta la información contenida en el servidor, no es obtenida directamente desde la base de datos sino a través de los servicios que expone dicha información.

Otro conjunto lo conforman aquellos componentes que se ven relacionados a través de la interfaz de usuario. Esto ocurre en el caso de la emisión en vivo, donde la transmisión y consumo de audio y video se realiza en forma totalmente independiente al chat. Estos componentes están desacoplados, pero ligados en la interfaz de usuario para que, de esta forma, puedan emitir o ver una entrevista y compartir mensajes con otras personas como si de una única funcionalidad se tratase.

El último grupo de componentes son los que cuentan con una fuerte relación, por ende, con un acoplamiento definido, y por ello se dice que están fuertemente acoplados. Se verá la comunicación o relación de ellos más adelante.

En cualquiera de los casos antes mencionados el Framework .NET actúa de base e infraestructura para la resolución en forma efectiva de los distintos escenarios planteados.

El fruto de la unión de todos estos componentes es la construcción de una aplicación 'cliente inteligente', también llamada *Smart Client*, comunicándose a través de servicios Web en una arquitectura cliente-servidor.

Dado el grado de importancia que adquiere la interfaz gráfica del producto final, los posibles cambios continuos en la información que éste maneja y la separación en los grupos débilmente acoplados lograda, se decidió aplicar el patrón Modelo-Vista-Controlador (MVC) [ver anexo A2-2. Modelo-Vista-Controlador]. El cometido es reflejar y distinguir entre presentación y lógica de la aplicación. La lógica, según la parte del sistema en cuestión, caerá en alguno de los grupos descritos, mientras que la presentación quedará independiente de éstos.

Como resultado de aplicar el patrón antes mencionado, respecto a la 'Vista' se dispone de un conjunto de paneles, definidos en un grupo menor de pestañas, pudiendo navegar entre ellos mediante los botones que contienen. Las pestañas están insertas en una ventana principal de forma tal de proveer al usuario una interfaz cómoda de trabajo, tanto por el aprovechamiento máximo del tamaño de la interfaz, como de la facilidad de intercambiar la pestaña en uso.

Una de las ventajas de usar Windows Presentation Foundation (WPF) es aprovechar al máximo la potencia que éste brinda para el desarrollo de ricas interfaces de usuario [ver anexo A2-3. Windows Presentation Foundation (WPF)].

La interfaz desarrollada está compuesta por controles, como ser diferentes tipos de contenedores, controles para añadir pestañas, grillas, botones, campos de texto, etiquetas, listas de valores, visores de documento. Los controles a su vez cuentan con eventos asociados.

Estos eventos componen la capa 'Controlador', a través de ellos se accede a los servicios prestados por la capa Modelo del patrón aplicado.

La capa 'Modelo' del MVC se puede dividir en subsistemas que ofrecen determinada funcionalidad y que juntos conforman la solución completa:

- Gestión de base de datos.
- Capa de acceso a datos.
- Chat.
- Emisión.
- Publicación y descarga de archivos.

4.2.1 Gestión de base de datos

Toda la información producida por la solución debe ser almacenada en un esquema de base de datos relacional alojada en el nodo servidor. Además se mantiene en cada cliente una base de datos local con información parcial del sistema. Debido a que la solución es un *Smart Client* que cuenta con funcionalidades que están disponibles aunque el sistema no esté conectado es necesario mantener réplica de parte de la información en forma local para que el sistema pueda reproducir las entrevistas descargadas. No se replica absolutamente toda la información sino que solamente se descarga información de las entrevistas que son descargadas.

Base de datos servidor

En el esquema relacional Microsoft SQL Server se mantiene la información referente al sistema, en una base de datos llamada *Gesell*. La estructura de esta base se describe en el anexo [ver anexo A3-1. Base de datos]. Los datos disponibles en tablas y vistas serán accedidos por la aplicación a través de la capa de acceso a datos, más específicamente, a través del contexto de datos de Linq to SQL.

Base de datos local

La información asociada a cada una de las entrevistas que el usuario ha descargado es almacenada localmente en su equipo, manteniendo la información necesaria a ser usada por el sistema [ver anexo A3-3. Componente local].

Con el fin de gestionar dichos datos locales se mantiene en cada cliente una base de datos instalada sobre Microsoft SQL Server 2005 Compact Edition que contiene parte de la información almacenada en el extremo servidor.

4.2.2 Capa de acceso a datos

Esta capa es la encargada de resolver el acceso a la base de datos llamada *Gesell* de SQL Server. La arquitectura utilizada para solucionar el acceso a datos consiste en dividir la capa en tres subcapas:

- Contexto de datos de Linq to SQL.
- Entidades.
- Servicios de WCF.

Todos los componentes de esta capa están instalados en el servidor de aplicaciones. En el mismo se expone solamente lo necesario a través de la subcapa de 'Servicios de WCF'. Los datos a los clientes se envían serializados, como representaciones XML de las entidades o como objetos proxy. En la subcapa de servicios, también se agrega lógica de negocios. Los datos que se envían al cliente se obtienen a partir de las clases entidades, que no son 'visibles' desde la aplicación cliente-remota. Es en estas clases y a través del contexto de datos de Linq to SQL que se accede a los datos de la base.

Se muestra en la siguiente imagen la arquitectura propuesta.

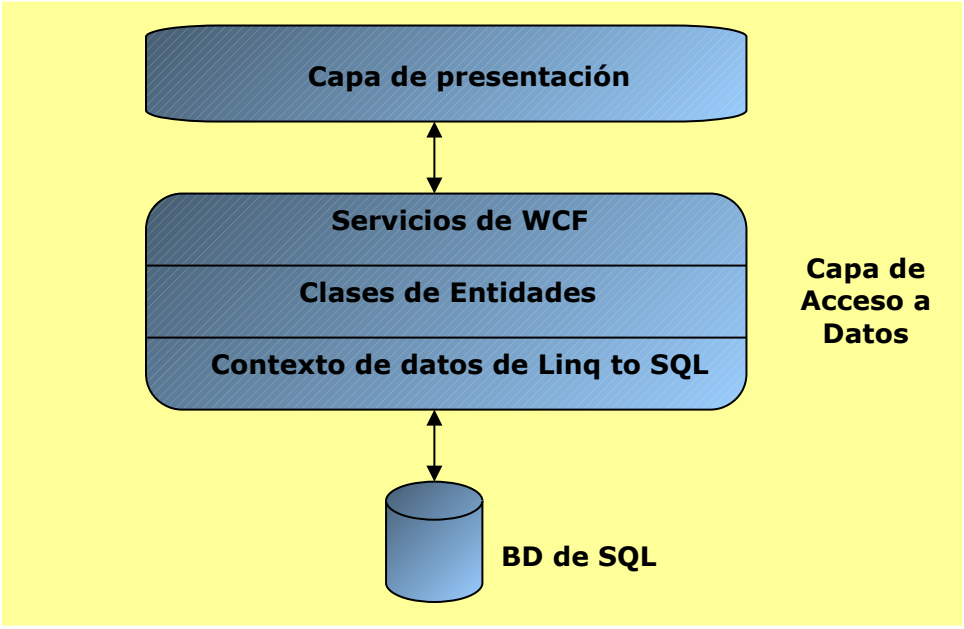


Figura B4.1:

Arquitectura de acceso a datos.

4.2.3 Chat

Este componente les permite a los usuarios intercambiar opiniones durante el transcurso de la entrevista.

Un punto importante que debe ser tenido en cuenta al trabajar en esta componente es la propia infraestructura de red, donde deberá ser necesario mantener una fuerte independencia entre su topología y la solución diseñada. Es así que, sacando partido a la característica del enfoque apuntado a servicio, donde se aprovecha la cualidad estándar del protocolo, es posible atravesar firewalls y routers sin necesidad de configuraciones extra, alcanzando de esta forma la independencia buscada.

Para cumplir con este requisito y lograr una solución independiente del entorno e infraestructura disponible se dispuso de un servicio alojado en el servidor de aplicaciones. Dicho servicio ofrece a la aplicación cliente las distintas funciones necesarias para implementar la sala de chat.

En forma paralela, en el equipo del usuario emisor de la entrevista, para cada conversación se genera un archivo con formato XML, conteniendo toda la información de los mensajes cursados, que se agrega a la entrevista correspondiente para luego ser publicada en el servidor y compartida con la comunidad.

En la siguiente figura se ilustra la forma en que se ha modelado este subsistema, recordando que el servicio implementado es un componente más del servidor de aplicaciones ya disponible.

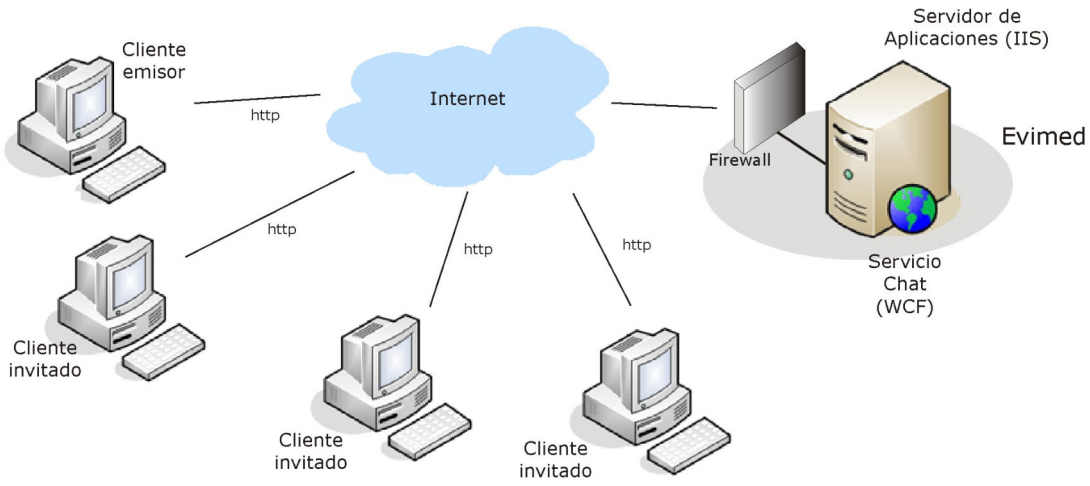


Figura B4.2: Arquitectura de subsistema de chat.

4.2.4 Emisión

Uno de los requisitos más importantes que debía cumplir la solución creada era brindar a los usuarios la posibilidad de compartir audio y video en vivo a través de Internet. De esta forma, le permitiría a un conjunto de invitados poder presenciar la entrevista en el mismo momento que la misma se desarrolla.

La herramienta capaz de resolver este requerimiento es una aplicación que sirve contenidos multimedia vía red, comúnmente llamada Servidor de streaming, por medio de la cual es posible emitir acontecimientos en tiempo real [ver anexo A4-4. Servidor de streaming].

El producto Windows Media Services9 de Microsoft instalado en un equipo servidor ejecutando Windows Server 2003 fue el seleccionado para cumplir esta función.

Hablando en términos de red, en la mayoría de los servicios de streaming es común que el productor de contenido esté lo más cerca posible del servidor. En este modelo el productor de la señal nunca está en el mismo sitio que el servidor por lo que el flujo producido debe ser enviado a través de Internet con las consecuencias que esto trae. Debido a las restricciones en el desarrollo de las componentes y dentro de las distintas alternativas de configuración con que cuenta esta aplicación servidor, se optó por implementar una solución donde este flujo sea inyectado desde el cliente con el protocolo de transporte TCP, más precisamente HTTP.

Por la naturaleza distribuida de la solución, esto es clientes remotos que se conectan vía Internet a un nodo central, se implementó el servicio con comunicación unicast y conexión vía UDP si el entorno así lo permitiese [ver anexo A4-4. Servidor de streaming].

De esta forma, la aplicación cliente en su rol de emisor (encoder) autenticada en el servidor de streaming, define un punto de publicación único para cada entrevista, donde irá dejando a través de la red el flujo de datos multimedia conforme estos se van generando. Los clientes que consuman este flujo deberán apuntar directamente a la dirección URL específica que contendrá el punto de publicación correspondiente.

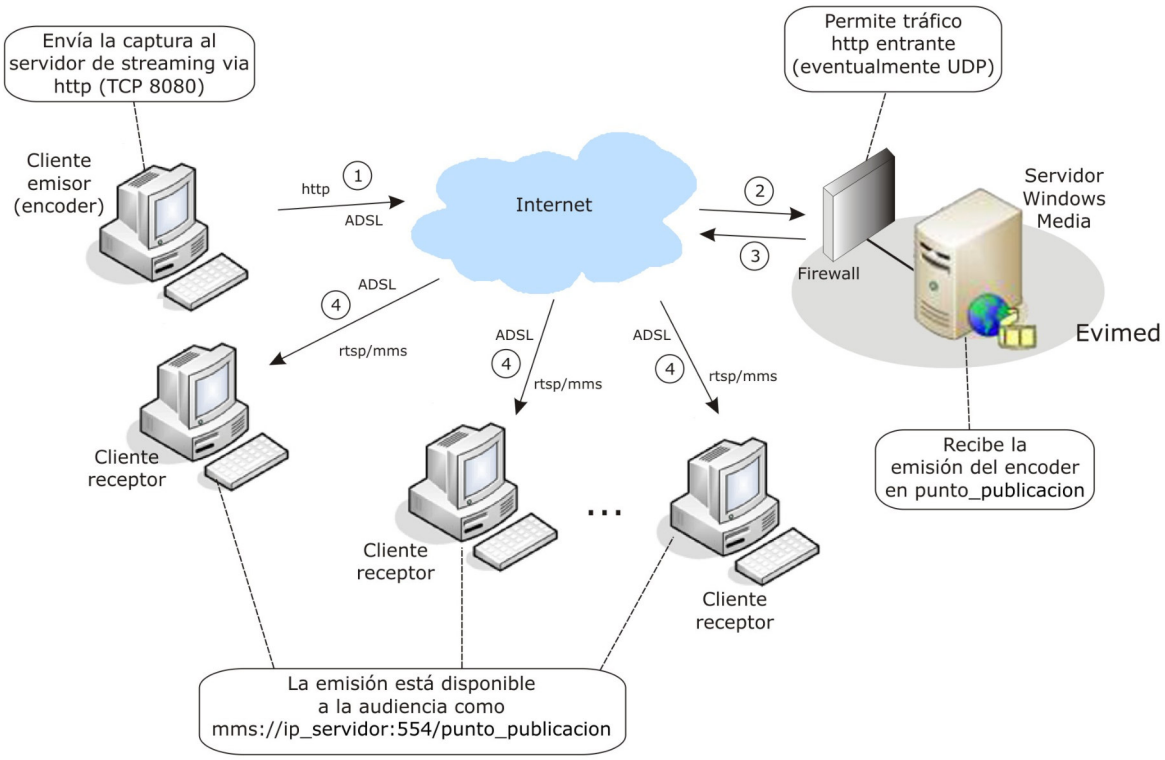


Figura B4.3: Arquitectura de subsistema de emisión.

4.2.5 Publicación y descarga de archivos

La sección de descarga y consumo a demanda de las entrevistas ya emitidas, así como la publicación de documentos de interés, requieren la posibilidad de que los usuarios sean capaces de transferir archivos en ambas direcciones. Este subsistema se encarga de cubrir cualquier necesidad de transferencia de archivos que se deba realizar en el sistema.

Al igual que el subsistema de chat, este componente también debe tener independencia del entorno de red, por lo cual es implementada a través de un servicio Web WCF aprovechando así la facilidad que HTTP brinda para atravesar firewalls y routers [ver anexo A2-4. Introducción a Windows Communication Foundation (WCF)].

Como en la mayoría de los casos los datos transferidos son contenidos multimedia, es necesario ajustar algunos parámetros en este servicio Web. Tanto el tamaño de transferencia, como el tiempo de espera, de recepción y envío de mensajes deben modificarse para soportar el gran volumen de transferencia.

El servidor de aplicaciones instalado en el extremo servidor es quien hospeda este servidor de archivos.

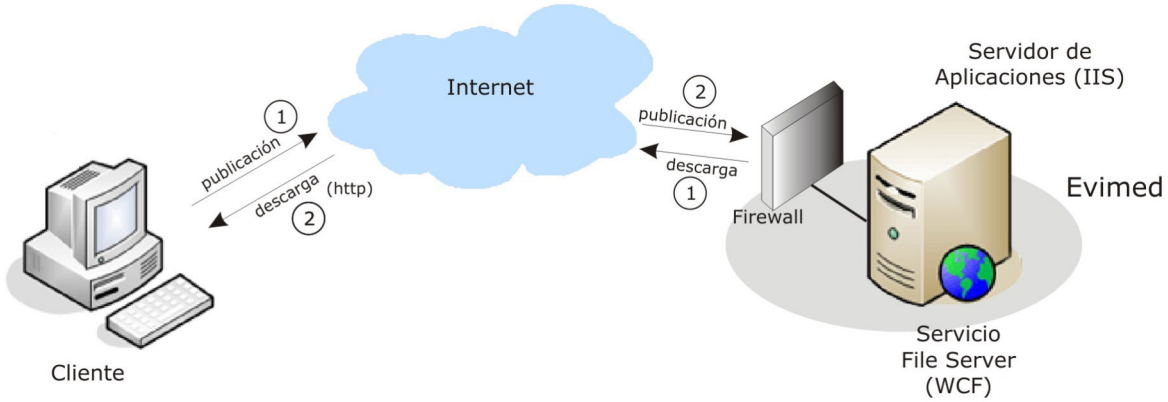


Figura B4.4: Arquitectura de publicación y descarga de archivos.

En resumen, cada subsistema antes descrito aporta funcionalidades específicas necesarias, con un cierto grado de acoplamiento entre sus componentes. La unión de los mismos conforma la arquitectura general de la solución.

Dicha solución consta de una aplicación cliente inteligente que obtiene y envía información por medio de servicios a través de Internet, lo que marca el carácter de arquitectura distribuida. En tal sentido se mantiene una base de datos local que le permite al cliente mantener una traza de la información con la que ya ha trabajado anteriormente.

La naturaleza del enfoque basado en servicios requiere la existencia de un nodo servidor mediante el cual se podrá enviar y recibir toda la información que el sistema necesite. Dicho extremo puede o no instalarse en un único equipo o máquina virtual y debe estar compuesto por un conjunto de aplicaciones servidores:

- Servidor de contenidos Windows Media Services (WMS v9), transmisión y distribución de entrevistas en vivo.
- Servidor de aplicaciones Internet Information Services (IIS v6), presentación de servicios Web desarrollados en WCF.
- Servidor de base de datos: SQL Server 2005, versión Express, almacenamiento de datos.

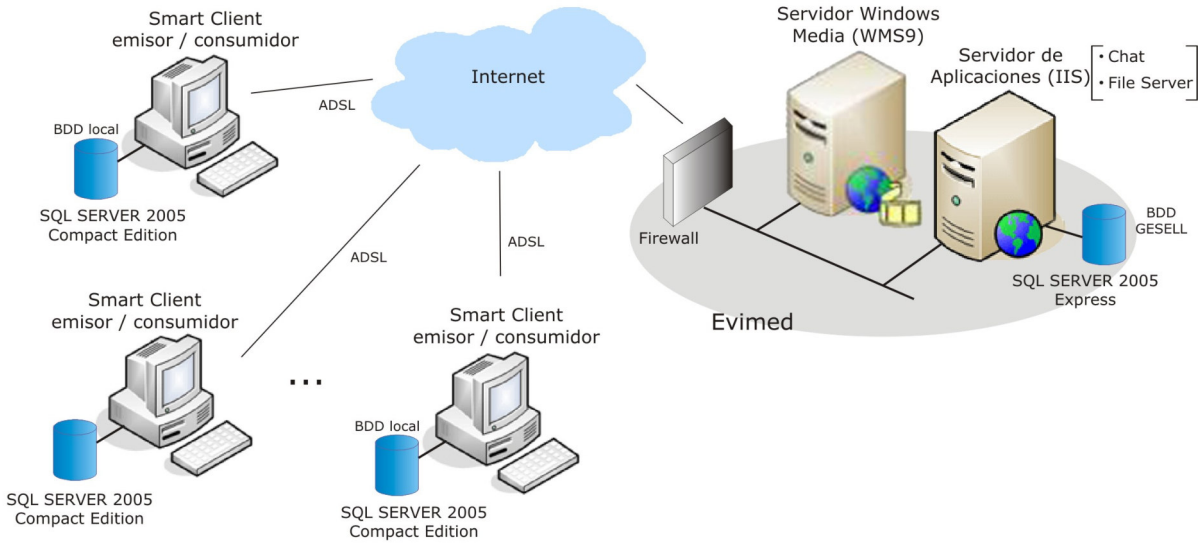


Figura B4.5: Arquitectura general de la solución.

En la figura B4.5 se resume en forma esquemática cómo los distintos subsistemas interactúan para conformar la arquitectura de la solución.

4.3 Diseño

Se divide el diseño de la aplicación a nivel de servidor y de cliente. Los componentes se agrupan teniendo en cuenta aquellos que tienen funcionalidades similares. Todos los componentes que realizan consultas sobre la base de datos del servidor conforman un gran componente llamado 'capa de acceso a datos', pero cada uno de ellos se encarga de la actualización de alguna de las entidades por separado. De igual forma, todos los componentes que proveen acceso a la información del servidor a través de servicios web conforman un gran componente denominado 'capa de servicios de acceso a datos', donde cada componente brinda información por separado de las entidades asociadas.

4.3.1 Servidor

En el extremo servidor debe resolverse la gestión de la información que se irá acumulando durante el uso del sistema. Esta información será consumida por todos o un grupo de clientes, por eso debe estar disponible en un lugar común a todos los usuarios.

Debe diseñarse una forma de poder acceder a los datos disponibles en el servidor en forma local al mismo servidor. Pero también una forma en que los clientes puedan consumir esta información pero no directamente desde los datos. Debe resolverse de forma independiente que el servidor exponga a los clientes la información estrictamente necesaria, ocultando el conjunto de datos que no puede estar disponibles para usuarios finales, sólo para el correcto funcionamiento del sistema.

La funcionalidad principal del sistema consiste en poder visualizar entrevistas en vivo emitidas por cualquiera de los usuarios y consumida por algún grupo de usuarios predefinidos. Estas emisiones van acompañadas por la posibilidad de intercambiar mensajes entre los usuarios, durante la duración de la entrevista. Esta parte del diseño agrupa un porcentaje interesante de las funcionalidades del sistema.

Datos

La información generada por el sistema consiste en datos asociados a entrevistas, usuarios, publicaciones, entre otros. Estos son mantenidos en una base de datos. Las entrevistas en vivo generan información conformada por videos con audio, imagen previa e información de los mensajes intercambiados por los usuarios. Estos se mantienen en diferentes tipos de archivos.

- **Base de datos**

La información de *G-Sell* se mantiene a través del sistema de gestión de base de datos relacionales Microsoft Sql Server 2005 Express. Éste permite la creación de sistemas sólidos y fiables debido a la robustez del sistema de base de datos. Cuenta con un optimizador avanzado de consultas, importante para resolver en tiempos mínimos las consultas, tratando de reducir el tiempo de respuesta de la aplicación remota. Maneja diferentes niveles de seguridad para el acceso a los datos, contando entre otros con compatibilidad con la autenticación de Windows.

Como la versión Express de Sql Server no cuenta con interfaz gráfica para su gestión, se utiliza la herramienta Management Studio Express Edition. Esta herramienta permite gestionar las bases de datos y sus componentes de forma gráfica.

Se crea una base de datos llamada *Gesell* a la que se accede a través de autenticación de Windows. Para esto se define un usuario en el servidor de base de datos, con permisos de lectura/escritura sobre la base *Gesell*.

Las entidades que se mantienen en la base de datos son: usuarios, especialidades de usuario, publicaciones, entrevistas e información de metadata de las entrevistas. Se tiene también información complementaria acerca de definición de entidades, configuración del

sistema y metadata asociada a cada especialidad. Para relacionar entidades se tienen relaciones entre usuarios y especialidades, y, entre entrevistas y usuarios invitados. Además, de forma auxiliar para el chat de las entrevistas en vivo se tienen las entidades usuarios conectados y mensajes de las entrevistas. Se definen algunas vistas auxiliares para ser usadas por el componente que accede a los datos [ver anexo A3-1. Base de Datos]. Se definen en el anexo también, las estructuras de las tablas de la base de datos *Gesell*. En forma gráfica se pueden observar las tablas definidas en la base y sus relaciones.

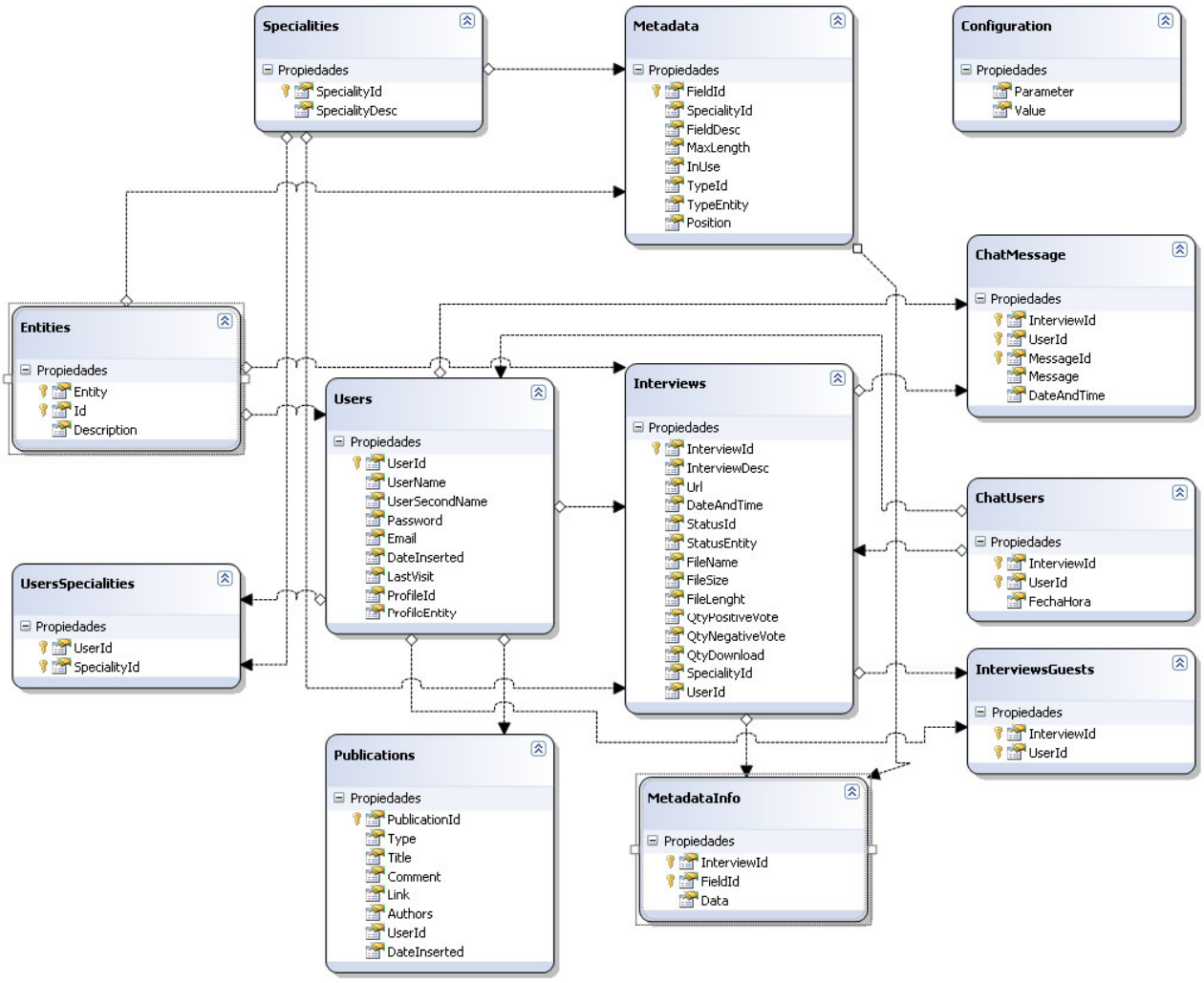


Figura B4.6: Tablas de la base de datos Gesell.

- **Archivos**

Finalizada la entrevista, el equipo cliente emisor genera tres archivos que luego son publicados en el servidor.

En la base de datos central se mantiene un parámetro identificado como ServerUpload que define el camino absoluto de la carpeta usada por el servicio Web para almacenar, en el servidor, todas las entrevistas producidas por el sistema.

De manera de organizar la transferencia, los archivos son empaquetados juntos con extensión zip y guardados, por el servicio Web que se encarga de las transferencias, en la carpeta determinada por el parámetro ServerUpload [ver anexo A2-5. Publicación y descargas de archivos].

En esta carpeta destino del servidor de archivos se mantienen dos entradas por cada entrevista publicada, por un lado la imagen de su portada y por otro la información empaquetada en sí misma.

La decisión de duplicar la imagen de portada -se encuentra además en el archivo empaquetado- se debe al constante uso que el sistema realiza de las portadas.

Es necesario además, que el directorio correspondiente a este servicio Web tenga permisos de escritura y modificación para el usuario que ejecuta el servidor de aplicaciones (IIS). Los permisos deben ser asignados tanto a nivel de la aplicación servidor como del sistema de archivos del propio Windows (NTFS Windows Server 2003).

Capa acceso a datos

En esta capa se resuelve el acceso a los datos de la base *Gesell* de Sql Server. Como se vio en la definición de la arquitectura del sistema, se divide la capa en tres subcapas:

- Contexto de datos de Linq to Sql.
- Entidades.
- Servicios de WCF.

Contexto de datos de Linq to Sql

Linq to Sql permite administrar los datos relacionales como objetos, asigna al modelo de base de datos un modelo de objetos. En tiempo de ejecución, Linq to Sql resuelve las consultas Sql a ser enviadas a la base de datos a partir del modelo de objetos. Una vez ejecutada la consulta sobre la base de datos y obtenidos los resultados, el proveedor Linq to Sql es el encargado de convertirlos en una colección de objetos de usuario. De esta manera, los datos de la base de datos quedan mapeados al modelo de objetos.

Linq to Sql cuenta con una herramienta incluida en la IDE de Visual Studio llamada 'Diseñador relacional de objetos'. Una vez que el modelo se ha completado en el diseñador, es posible obtener código generado automáticamente por Visual Studio, a partir de su definición.

El código base generado automáticamente es el de las clases entidades, que luego podrán ser extendidas. Se genera para cada clase entidad una clase parcial, en donde los campos de cada tabla corresponden a propiedades de cada clase. Además se incluyen métodos de uso habitual y que son necesarios para poder trabajar con las instancias de las clases.

Se genera también automáticamente la clase de contexto de datos llamada *GesellDataContext*. Es a través de esta clase que las clases entidades realizan consultas sobre la base de datos. Esta clase contiene propiedades que representan las entidades que se modelan y métodos que representan los procedimientos almacenados de la base de datos. Cuando se crea una instancia de esta clase funciona como punto de entrada a la base de datos definida en la conexión.

Entidades

Las clases parciales generadas automáticamente se extienden mediante otras clases parciales por el programador. Se habla de 'clases parciales' porque una clase puede definirse en diferentes lugares en forma parcial. Pero el conjunto de estas clases parciales definen una única clase.

Básicamente, se extienden las diferentes clases con los siguientes métodos:

- Load: Para obtener datos de las entidades desde la base de datos.
- Save: Para realizar altas o modificaciones de entidades.
- Delete: Para eliminar entidades de la base de datos.

La clase *GesellDataContext* conforma la subcapa de contexto de datos de Linq to Sql. Las clases parciales generadas automáticamente a partir del 'Diseñador relacional de objetos', y las clases parciales desarrolladas para extender las funcionalidades de estas clases, conforman las subcapa de entidades.

Estas dos subcapas se incluyen dentro del componente *Gesell.DataAccessLayer* [ver anexo A3-2. Acceso a Datos].

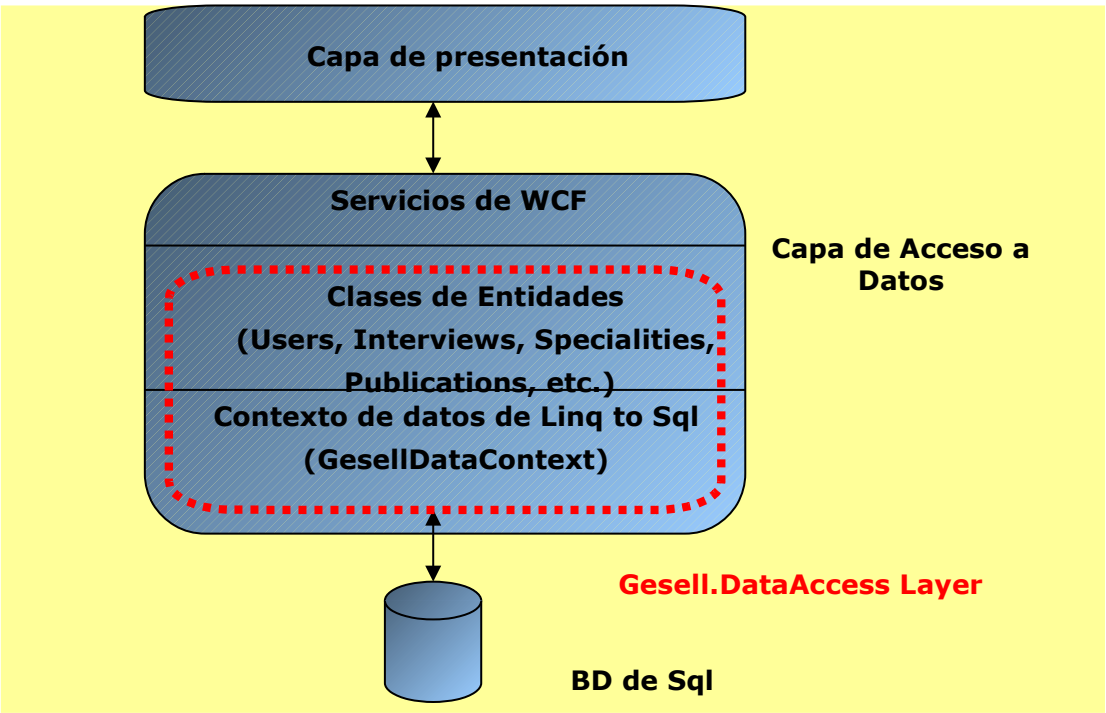


Figura B4.7: Componente de acceso a datos.

- **Servicios de WCF**

La parte de la arquitectura que se basa en servicios Web se resuelve a través de Windows Communication Foundation. Los servicios exponen funcionalidades desde el servidor de aplicaciones a los clientes inteligentes.

Se definen varias clases para implementar los servicios. Para cada clase entidad definida en la subcapa de entidades se define una clase dentro de los servicios. Estas clases exponen a la capa de presentación métodos a ser invocados. En estos métodos se recibe o envía información desde el cliente a la base de datos y viceversa. Esta información consta de parámetros de tipos básicos como ser strings, booleanos y numéricos o también pueden ser objetos estructurados. La estructura de estos objetos se define en el propio servicio. Es así como el servicio oculta información de la base de datos a los clientes, proporcionando la información estrictamente necesaria. Para esto se define en cada servicio contratos de servicio, de métodos y de datos.

La manera en que los servicios obtienen la información a ser expuesta es instanciando clases entidades. Con los datos que estas clases proporcionan, se crean los objetos que se retornan desde el servicio. En caso de estar enviando información a la base de datos, con los objetos recibidos en la invocación a un método de un servicio desde la presentación del cliente, se definen las propiedades de los objetos de las clases entidades. Más tarde, se invocan métodos de estas clases y es en esta subcapa donde se agrega lógica de negocios, ya sea cuando la información viaja hacia la base de datos desde el cliente, o viceversa.

El funcionamiento de los servicios de WCF puede verse detallado en un anexo [ver anexo A2-4. Windows Communication Foundation (WCF)]. Todas las clases que implementan estos servicios forman parte del componente Gesell.DataAccessServiceLayer.

Chat

La comunicación entre usuarios mediante el chat fue diseñada e implementada de dos maneras diferentes. Se contó con una tercera forma de resolver este propósito que no llegó a implementarse pero que también es una solución viable.

- **Servicios dúplex**

Se comenzó con una comunicación dúplex. Este tipo de comunicación utiliza servicios dúplex *wsHttpDualBinding* [ver anexo A2-7. Servicios dúplex]. Se trata de un modelo de intercambio en el que ambos extremos pueden enviar mensajes al otro de manera independiente. La comunicación se establece cuando el cliente se conecta a un servicio y le proporciona un canal en el que dicho servicio le puede devolver los mensajes. Debido a la forma dual de comunicación es necesario que cada extremo conozca la dirección del otro. Esta condición puede, en ciertos casos y entornos de red, transformarse en una pérdida de generalidad, dado que las direcciones finales de los clientes no siempre pueden ser identificadas en Internet.

- **Comunicación punto a punto**

Frente a la problemática de los servicios dúplex, se investigaron otras alternativas. Una de ellas fue Perchannel, comunicación punto a punto [ver anexo A2-8. Chat del canal del mismo nivel]. Esta opción no fue implementada debido a que se buscaba brindar la mayor generalidad posible respecto a entornos de red.

- **Servicios Web wsHttpBinding**

Otra de las alternativas estudiadas, elegida finalmente por ser una solución rápida al problema presentado, fue la de usar servicios Web comunes (wsHttpBinding), almacenando los datos en la base ubicada en el servidor. Este modelo de servicio Web también fue utilizado en el diseño del sistema para resolver otros requerimientos como el Acceso a Datos, entre otros.

Como se expuso en la sección Arquitectura, resulta importante alcanzar cierta independencia entre la solución planteada y el entorno de red disponible, sobre todo considerando la diversidad de alternativas con que cuentan los usuarios finales para conectarse a Internet.

Es por esto que se resolvió en definitiva utilizar los servicios web para implementar una cola de mensajes asociados a la entrevista. De esta manera, cuando un usuario accede a la emisión de una entrevista, ya sea como emisor o como receptor, a través de los servicios se guarda la información del usuario en una tabla de la base de datos. Cada vez que un usuario envía un mensaje a la entrevista, éste también se guarda en una tabla de la base. Además del mensaje se registra información de fecha y hora en la que se recibe por lo que se genera una cola de mensajes. El sistema consume regularmente mensajes de la cola de manera de generar el servicio de chat, actualizando la interfaz con los nuevos mensajes recibidos.

Streaming

La difusión de una emisión en el momento que está ocurriendo se lleva a cabo mediante una herramienta que oficia de servidor de contenidos multimedia o servidor de streaming [ver anexo A4-4. Servidor de streaming].

Dentro de las alternativas que el servidor tiene para distribuir contenido, se opta por usar la capacidad que tienen los clientes de cambiar de protocolo de forma inteligente según el entorno en el que se encuentran. De esta forma intentan primero con los protocolos apropiados para streaming sobre UDP antes de hacerlo en TCP.

Para la inyección del contenido multimedia se realiza una transmisión vía HTTP, más precisamente TCP a través del puerto 8080. Aunque en ciertas condiciones no es la solución recomendada para streaming en tiempo real a través de Internet, debido al control de flujo y sobrecarga que aporta TCP, esta solución fue la escogida por restricciones encontradas en la implementación de las librerías que se utilizan para resolver la comunicación con el servidor de streaming [ver anexo A4-1. Captura y streaming multimedia].

De esta forma, el cliente emisor (encoder) se define un punto de publicación de difusión configurado por defecto en el servidor donde inyecta ('push') en tiempo real a través de TCP el flujo multimedia generado.

El servidor de streaming está configurado para que únicamente cierto grupo de usuarios del sistema Windows instalado en dicho equipo servidor sea capaz de crear puntos de publicación y transmitir. En tal sentido, se define un usuario genérico perteneciente a este grupo que es usado por el cliente emisor para autenticarse y efectuar la transmisión.

La herramienta seleccionada como servidor de contenidos es Windows Media Services9 de Microsoft instalada bajo Windows Server 2003, que cuenta con una interfaz de administración donde se puede detener, reiniciar y monitorear el servicio, además de publicar, cortar, bloquear e incluso eliminar puntos de publicación.

Debido a que las tecnologías de streaming utilizadas se valen de la técnica de buffering para lograr fluidez y una buena experiencia de reproducción, se agrega un retraso a la emisión en vivo que, dependiendo del caso, puede alcanzar los 20 segundos. Desde el punto de vista de la audiencia, por tratarse de una solución pesada para funcionar a través de Internet, no es posible aprovechar las bondades que brindan las transmisiones multicast, por lo que se debió configurar la comunicación en forma de unicast. En estas condiciones, se restringe la cantidad de usuarios concurrentes según el ancho de banda disponible y la potencia del equipo servidor [ver anexo A4-4. Servidor de streaming].

4.3.2 Cliente

En el extremo del cliente debe gestionarse la información referente a las entrevistas que los usuarios descargan en el disco local. Esta información solamente es usada en el ámbito del equipo cliente. Debe resolverse entonces el acceso a estos datos para que la aplicación pueda usarlos en las diferentes interfaces de usuario.

Las funcionalidades más relevantes serán las que permitan visualizar los videos descargados junto con los mensajes que hubieran sido intercambiados en el transcurso de la misma. Además de todo esto, la aplicación cliente deberá manejar el área para publicaciones de artículos de interés por lo que se apunta a estrechar el vínculo entre esta y las aplicaciones Web que brindan la información. Esto motiva el diseño y construcción de un cliente inteligente (Smart Client) que permite aprovechar la tecnología XML para obtener lo mejor de ambos lados, combinando el alcance de Internet con la potencia del hardware local.

La tecnología de aplicaciones Smart Client resulta ser la indicada para la solución diseñada puesto que permite a los usuarios trabajar con información aún cuando no están conectados a Internet, ya que parte de la lógica se encuentra en el cliente [49].

Datos

Los datos que se deben guardar localmente son los asociados a las entrevistas que los usuarios descargan en sus equipos. Estos son una parte de los datos gestionados en el servidor de base de datos, los que son relevantes a la entrevista. Otros son los archivos que contienen el video, la imagen previa, y los mensajes de una entrevista.

- **Base de datos local**

Se utiliza para gestionar los datos locales en cada cliente del sistema Microsoft Sql Server 2005 Compact Edition. Se cuenta con una base llamada *GesellLocal.sdf*. Básicamente contiene un subconjunto de información de la base remota *Gesell* de Sql Server 2005 Express.

Algunas de las tablas de la base *GesellLocal.sdf* tienen la misma estructura que la base *Gesell,* mientras que otras están compuestas por algunos de los campos. Además se cuenta con una tabla que sólo se usa de manera local. Otro grupo de tablas de *Gesell* no son incluidas en la base local [ver anexo A3-3. Componente local].

La información almacenada en esta base corresponde a datos y metadatos de la entrevista, usuario que la publicó, usuario que la descargó, además de especialidades asociadas a los usuarios y entrevistas.

- **Archivos locales**

En el momento que se da por finalizada una emisión se generan en el disco local del cliente emisor, más precisamente en la carpeta Emisiones bajo el directorio de instalación del sistema, tres archivos que contienen toda la información correspondiente a dicha emisión.

Estos archivos son los que la aplicación desempeñando el rol emisor deberá empaquetar y publicar en el servidor al cerrar la entrevista, para que luego sean descargados y reproducidos a demanda por los usuarios autorizados.

Debido a la sensibilidad de la información manejada, se encriptan los datos mediante una clave en el momento que están siendo empaquetados. Esto permite aportar un nivel de seguridad en el transporte de dicha información [ver anexo A2-5. Publicación y descarga de archivos].

Una vez descargada la entrevista en el equipo cliente, además de la información contenida en la base de datos, esta tiene asociados los tres archivos que permanecen almacenados localmente en la carpeta *Emisiones* dentro de *Download*, que está bajo el directorio de instalación del sistema.

Uno de los archivos, que tiene formato y extensión *jpg*, es la primera imagen capturada por la cámara que luego es usada como información de portada.

El segundo es un archivo, con formato ASF y extensión *wmv*, que contiene la grabación de audio y video producidos durante el transcurso de la entrevista [ver anexo A4-3. Formato de archivos multimedia].

Por último, se cuenta con un archivo formato XML que guarda la información referente a los mensajes intercambiados por los usuarios durante una entrevista. La lista de cada mensaje cursado conteniendo varias propiedades se guarda bajo el nodo raíz del archivo.

El formato genérico y la estructura de cada mensaje se pueden apreciar en la siguiente figura.

```
<?xml version="1.0" encoding="utf-8" ?>
<RootFilexmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<arrInterviewMessages>
  <InterviewMessage>
       <UserId>Identificador del usuario</UserId>
       <Message>Contenido del mensaje</Message>
       <InterviewId>Indentificador de la entrevista</InterviewId>
       <hh>Hora en que se recibió el mensaje en el usuario emisor</hh>
       <mm>Minuto en que se recibió el mensaje en el usuario emisor</mm>
       <ss>Segundo en que se recibió el mensaje en el usuario emisor</ss>
  </InterviewMessage>
  <InterviewMessage>
        <UserId>xxx</UserId>
        <Message>xxx</Message>
        <InterviewId>xxx</InterviewId>
        <hh>xx</hh>
        <mm>xx</mm>
        <ss>xx</ss>
  </InterviewMessage>
</arrInterviewMessages>
</RootFile>
```

Figura B4.8: Formato archivo XML.

Acceso a datos

Gesell.DataAccessLayerLocal es el componente encargado del acceso a datos locales.

Contiene una clase asociada a cada tabla de la base. Estas clases mantienen propiedades asociadas a los campos de las tablas y tienen los métodos necesarios para actualizar y obtener información de cada una de las tablas. El componente cuenta además con

una clase que centraliza la lógica de la actualización de la base de datos local cuando ocurren descargas de entrevistas.

Emisión en vivo

- **Chat**

Para resolver el intercambio de mensajes entre usuarios durante el transcurso de la entrevista en vivo, en un estilo de chat, se implementó un componente llamado *ChatMessages*. Se debe tener en cuenta que la entrevista puede comenzar antes del inicio de la emisión de video y puede finalizar luego de concluida dicha emisión. En estos lapsos, el chat es la única funcionalidad disponible en la entrevista, dado que el intercambio de mensajes esta disponible antes, durante y después de la emisión en vivo del video.

Este componente guarda todos los mensajes que se van recibiendo desde los usuarios que consumen la entrevista. La información que éste almacena es el mensaje propiamente dicho, el identificador de la entrevista a la que pertenece, el usuario que lo envía, además de la fecha y hora de enviado. Esta acción se realiza en forma local en la aplicación cliente que está realizando la emisión. En el momento que se da por finalizada la emisión en vivo, cerrando el chat, se encarga de crear con los mensajes guardados en memoria el archivo con formato y extensión XML.

Básicamente utiliza dos métodos para realizar su función *AddMessage* que guarda en estructuras locales del componente los mensajes que se van recibiendo y *Save* que vuelca estos datos almacenados en las estructuras locales al archivo XML, siempre localmente en el equipo que emite la entrevista.

- **Multimedia**

Realizar una emisión en vivo de audio y video implica, además de la necesidad de contar con la infraestructura necesaria en el extremo servidor, dotar a la aplicación cliente de la capacidad de manejar flujos de información multimedia.

Con el fin de alcanzar este cometido se han seleccionado las librerías de DirectX, más precisamente DirectShow que proporcionan el control y manejo multimedia para el desarrollo de aplicaciones en sistemas Microsoft Windows [ver anexo A4-2. Multimedia en Windows].

Por tratarse de librerías para desarrollo que están diseñadas e implementadas en lenguaje C, se debió recurrir a un proyecto abierto y libre publicado en CodePlex denominado DirectShow.NET que es una implementación de una versión de estas librerías para .NET (C sharp).

La librería DirectShow permite programar en código de alto nivel el comportamiento y flujo de datos multimedia por medio de lo que se denomina grafo de filtros o '*filter graph*'. Esta estructura consta de distintos tipos de filtros que al ser conectados representan el modelo para tratamiento de información multimedia en aplicaciones Windows.

Ante cierto inconveniente en el tratamiento que WPF hace en la etapa de entrega para DirectShow (video en pantalla) y gracias al aporte del nuevo diseño en Direct3D se encontró el proyecto denominado MediaKit de Jeremiah Morril [ver anexo A4-2. Multimedia en Windows].

Esta librería sirve para construir en forma rápida y sencilla controles WPF en DirectShow obteniendo máximo provecho, con la aplicación de estas nuevas tecnologías, al momento de presentar la señal al usuario final.

Los controles incluidos en el paquete original no cubren ni la captura de audio ni la publicación en el servidor de streaming, por lo que debió ser modificado para adaptarlo a las necesidades propias de la solución.

También se cuenta con un modulo de autenticación y validación de credenciales contra el Windows Media Services (WMS) que permite a la aplicación interactuar con el servidor para emitir las entrevistas y así cumplir con las restricciones de seguridad impuestas.

Al emitir una señal en vivo se debe grabar el resultado en disco local y hacer uso del entorno de red, en particular Internet, para que el encoder pueda alcanzar el servidor de streaming y así inyectar el flujo generado. Es así que la calidad y continuidad del video obtenida al emitir una señal se ve afectada por el ancho de banda disponible.

Debido a la muy baja calidad de video obtenida para alcanzar un nivel de emisión aceptable, en las sucesivas pruebas realizadas se optó por mantener dos calidades de video en forma simultánea. De esta forma, se logra realizar una emisión en vivo aceptable y permitir la posibilidad de una mejor calidad cuando la entrevista fue descargada y se intenta reproducir localmente.

Microsoft cuenta con un conjunto de librerías WindowsMedia SDK pensadas para lenguaje C que se encargan del formato, grabación o envío por red de flujos multimedia. En forma similar a DirectShow, se debió utilizar la librería WindowsMediaLib.NET versión abierta y libre para .NET publicada también en CodePlex.

El formato de archivo multimedia escogido para grabar la captura a disco local es ASF de Windows, en particular Windows media video (WMV8) que por motivos de estandarización y restricciones en la implementación se prefieren ante la siguiente y última versión del formato (WMV9) [ver anexo A4-3. Formato de archivos multimedia].

Por último, se considera importante destacar que aunque el '*Picture in Picture*' no fue implementado, se consideró en la etapa de diseño. Es así que, en el momento de escoger la herramienta que resuelve la funcionalidad de tratamiento multimedia se optó por utilizar directamente las librerías DirectShow. Esta decisión permite, llegado el caso y con los filtros de transformación adecuados, manipular el flujo multimedia a nivel de *frame* logrando de esta manera la posibilidad de unificar dos flujos en uno.

Consumo de contenido

- **Chat**

En la emisión de entrevistas a demanda, el componente lee el archivo XML, lo carga en memoria en sus estructuras locales y retorna la información que contiene a la interfaz de usuario. De acuerdo a la información de fecha y hora de los mensajes, se irán desplegando en la interfaz de usuario cronológicamente durante la emisión del video, en forma paralela.

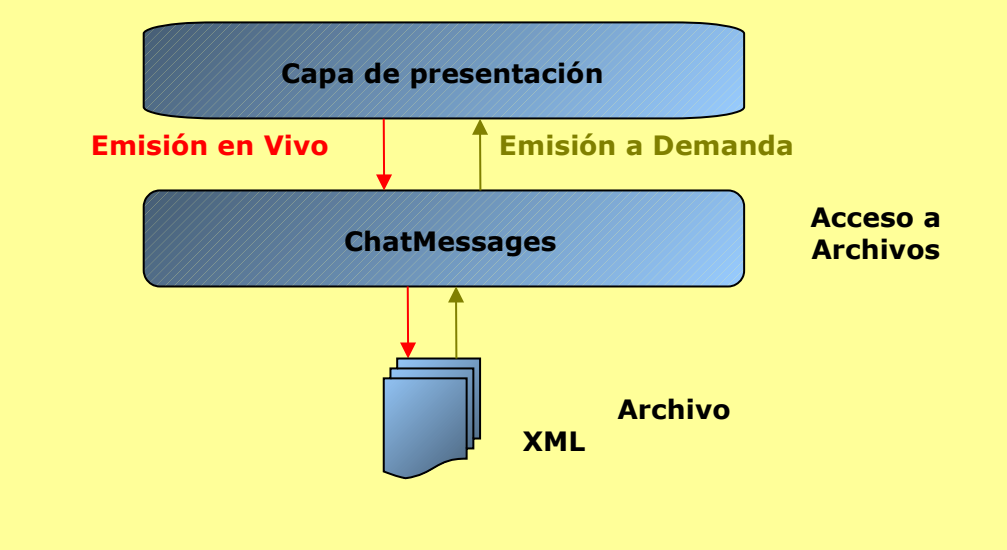


Figura B4.9: Acceso a archivos.

- **Multimedia**

Tanto para el consumo de contenido en vivo, como para la reproducción en forma local de las entrevistas se utiliza el control *MediaElement* provisto en forma estándar por WPF. Dicho control permite de manera sencilla, rápida y eficiente presentar contenido multimedia al usuario final haciendo uso del potencial que WPF aporta en la capa de presentación.

En el caso de consumo a demanda, el progreso del video reproducido debe estar sincronizado con la evolución del chat correspondiente.

Presentación

La interfaz de usuario juega un papel predominante en toda la aplicación. Se busca una interfaz lo más amigable posible para que los usuarios aún inexperientes en este tipo de sistemas, se vean tentados y motivados a su uso, y no desalentados por cuestiones de usabilidad. Se trata de una aplicación que sin conexión a Internet, puede ser usada por los usuarios localmente para ver las entrevistas que han sido descargadas previamente.

Si el equipo del usuario se encuentra conectado a Internet, según el rol que le fue asignado al usuario, podrá navegar por las distintas funcionalidades que la aplicación provee. Por más detalles remitirse a manual de usuario [ver Manual de usuario del sistema G-Sell].

Se debe destacar la aplicación de patrones básicos para el desarrollo de aplicaciones Web 2.0 [ver anexo A2-1. Ocho patrones básicos para el desarrollo de la Web 2.0]:

1- Aprovechar la inteligencia colectiva.
La aplicación de este patrón se ve reflejado en la posibilidad que se les brinda a los usuarios de agregar valor en forma directa mediante la participación del chat, comentando el caso, compartiendo información y conocimiento. Se produce de esta manera el llamado 'efecto de red', incrementándose éste con la posibilidad que la aplicación provee de invitar tantos usuarios a la entrevista como se desee, maximizando así los efectos de la participación individual. La aplicación se ejecuta sobre Internet, si la aplicación funcionara sólo sobre una red privada, estaría limitada a ese grupo de usuarios. Desde la perspectiva del 'emisor' la recepción de comentarios en una entrevista le aporta nuevos puntos a tratar con su paciente y sugerencias que sus colegas le hacen saber. La aplicación además, permite crear portadas con datos relevantes de la entrevista, del paciente y la especialidad, entre otros que proporcionando un marco amplio para la creación de contenidos. La confidencialidad de los datos que se comparten es responsabilidad exclusiva del usuario.

2- Los datos son el nuevo "Intel Inside".
La información que se maneja en cada entrevista además de sensible, es rica en cuanto a conocimiento. Al igual que en la Wiki se trata de información difícil y costosa de generar, pero a diferencia de esta no sólo consiste en la elaboración de un *artículo*, sino que agrega todo el conocimiento creado en la entrevista emitida en vivo. Se logra así obtener información exclusiva, transformando al dueño del servicio en poseedor de una fuente de datos única y difícil de recrear.

3- Experiencia rica a nivel de usuario.
El uso de un Smart Client proporciona una muy buena experiencia tanto en forma conectada, como desconectada. El usuario solamente deberá instalar en su equipo una aplicación liviana y de fácil uso, donde podrá navegar por diferentes pestañas obteniendo todas las prestaciones ofrecidas. Una de las funcionalidades a destacar es la posibilidad de realizar búsquedas sobre las publicaciones expuestas.

4- Beta perpetuo.
El usuario será el propio tester y es quien va a sugerir la incorporación de nuevas funcionalidades o cambios en la interfaz.

5- Aprovechar la larga cola (Long Tail).
La aplicación puede ser ejecutada ya sea en una red local como también en Internet, por lo que no tiene limitaciones y posee todas las ventajas que nos ofrece

Internet. Se diseñó un sistema de votaciones y ranking de las emisiones más descargadas, lo que guía a intereses personales y especializados.

6- Modelos livianos y escalabilidad rentable.
Es una aplicación escalable, se puede decir que sólo necesita un buen ancho de banda lo cual es cada vez más accesible. Es posible aplicar la solución a múltiples industrias por tratarse de una aplicación ampliamente configurable. La posibilidad de definir la metadata a ser manejada por el sistema hace que la información a ser asociada a las entrevistas pueda ser literalmente cualquiera. El significado de los datos será responsabilidad del usuario administrador del sistema, quien tiene la potestad de definirlos.

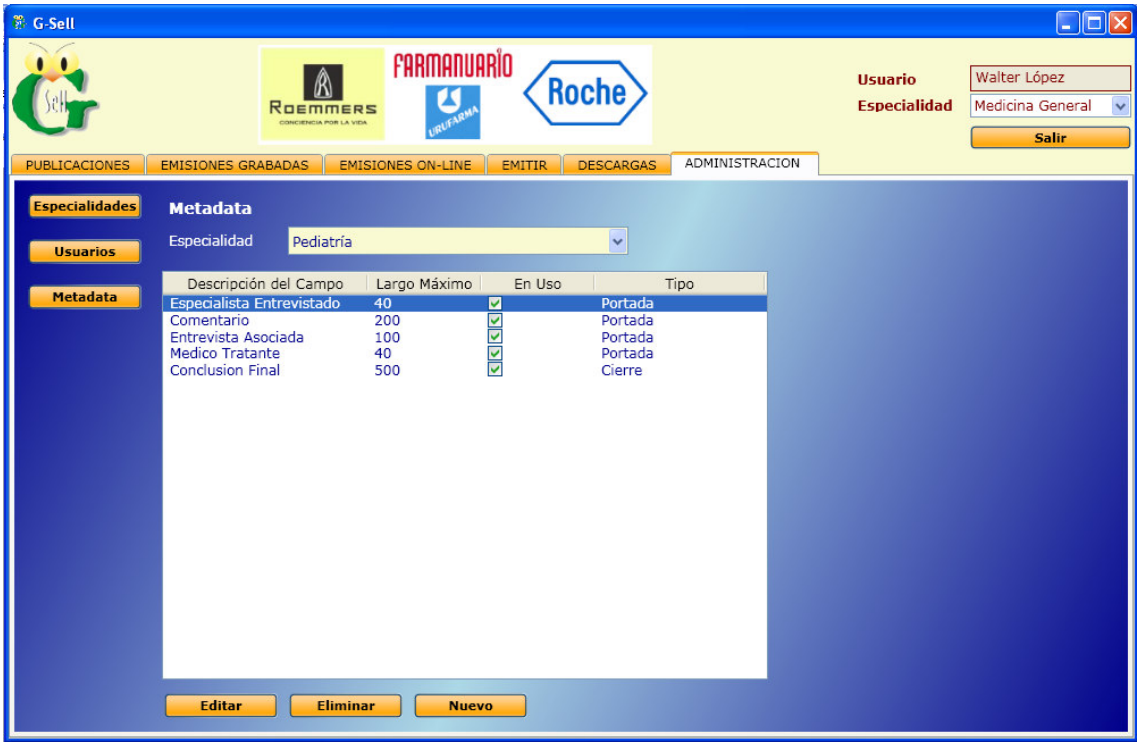


Figura B4.10: Definición de Metadata.

4.4 Requerimientos

Al tratarse de una solución *Smart Client* con arquitectura basada en comunicación por medio de servicios se requiere que cada elemento constituyente de la estructura posea requerimientos físicos diferentes.

En tal entorno resulta destacable señalar que la totalidad de las pruebas llevadas a cabo han sido realizadas en equipos con procesadores Intel (32bits), más de 1GB de memoria RAM y enlaces ADSL (1024/128Kbps).

Por la naturaleza cliente-servidor de la presente solución se distinguen dos clases de requerimientos para los equipos según la función que cumplan.

4.4.1 Servidor

En el extremo servidor deben instalarse los diferentes servicios que se ofrecerán a las aplicaciones clientes. Estos servicios (también denominados servidores) a su vez son aplicaciones que pueden o no estar en la misma máquina. Claramente, el ideal sería alcanzado si fuera posible disponer de varias máquinas físicas que cumplieran cada una la función de un servidor particular, inclusive si estuvieran en distintas máquinas virtuales. No obstante, es posible instalar todos estos servidores en una única máquina virtual y aún así lograr una buena

performance, debido a que la principal limitante del sistema es en ancho de banda y la infraestructura de red disponible.

Servidor de Base de Datos

Microsoft Sql Server 2005 Express Edition cuenta con tres versiones:

- Microsoft Sql Server 2005 Express Edition, version inicial.
- Microsoft Sql Server 2005 Express Edition Service Pack 1 y 2.

Los sistemas operativos compatibles con Microsoft Sql Server 2005 Express Edition son:

- Windows 2000 Service Pack 4.
- Windows Server 2003 Service Pack 1.
- Windows XP Service Pack 2.

Los requisitos de hardware para su instalación son:

- Procesador Intel Pentium (se recomienda 1GHz o más).
- 512MB de RAM o más.
- Tarjeta de red Ethernet.

Para poder aprovechar el tamaño máximo de la base de datos deberá contarse con 4GB más de espacio en disco, ya que es la máxima capacidad de almacenamiento de esta versión de Sql Server.

Como requisito previo de software deberá considerarse:

- Microsoft .NET Framework 2.0.
- Microsoft Internet Explorer 6.0 SP1 o posterior (requisito previo para .NET Framework).

En caso de contar con otros sistemas operativos, puede instalarse Service Pack 2 de Microsoft Sql Server 2005 Express Edition. En este caso los sistemas operativos compatibles son Windows Server 2003 Service Pack 1,Windows Small Business Server 2003 Service Pack 1, Windows XP Service Pack 2, Windows 2000 Service Pack 4, Virtual PC, Virtual Server, Windows Vista.

En el caso del sistema operativo Windows Vista, también es compatible con Service Pack 1 de Microsoft Sql Server 2005 Express Edition. Para las pruebas del sistema realizadas se ha instalado Microsoft Sql Server 2005 Express Edition sobre Windows Server 2003.

Servidor de aplicaciones

Por la propia característica de este servidor, el producto seleccionado como servidor de aplicaciones, comúnmente denominado servidor web, requerirá mayor potencia de hardware a medida que se vaya incrementando la carga del sistema. La solución Internet Information Services (IIS) versión 6.0 de Microsoft es la que se ajusta a nuestras necesidades y entorno.

La máquina que aloje este servicio necesitará satisfacer algunos requerimientos considerados básicos.

- Microsoft Windows Server 2003 o Microsoft Windows XP Professional.
- Procesador Pentium (se recomienda 1GHz o más).
- 256MB de RAM o más.
- Tarjeta de red Ethernet.

La cantidad de espacio necesario en el disco duro es el mínimo requerido para instalar correctamente el producto. Se deberá contar con cierto margen en espacio libre, de forma tal de crecer en la medida de ser necesario.

Todas las pruebas realizadas fueron ejecutadas en un IIS versión 6.0 bajo sistema operativo Windows Server 2003.

Servidor de transmisión de contenidos

El producto de la serie Windows Media Services9 es la opción escogida en la solución final como herramienta para oficiar de servidor de emisión de contenidos (servidor de streaming) y así transmitir las entrevistas en vivo a los clientes vía Internet.

Windows Media Services9 está disponible como componente adicional instalable para los sistemas:

- Windows Server 2003 (Standard, Enterprise, Datacenter).
- Windows Server 2008.

El equipo que contenga dicho servidor de contenidos deberá cumplir ciertos requerimientos mínimos:

- Procesador Pentium. (se recomienda 1GHz o más).
- 512MB o más de RAM es recomendado.
- Sistema de archivo NTFS.
- Tarjeta de red.
- Ancho de banda suficiente.

Es recomendable actualizar el servidor Windows Media con las últimas mejoras incluidas en el Service Pack 2 de Windows Server 2003, que permite incrementar tanto la seguridad como su rendimiento y fiabilidad. El ancho de banda en este caso juega un papel fundamental puesto que cuanto mayor sea, mejor será la performance del sistema de transmisiones en vivo.

Servidor de archivos

Los contenidos digitales generados por la aplicación en cada sesión emitida deben ser almacenados y estar disponibles en un sitio centralizado, donde todos los clientes autorizados puedan acceder para descargarlos.

La transferencia de archivos se realiza a través de un servicio Web especial, motivo por el cual el servidor de archivos puede estar instalado en el mismo equipo que contiene el servidor de aplicaciones. Por consiguiente, los requerimientos mínimos necesarios, salvando los de espacio en disco, implican tener instalado el servidor de aplicaciones.

- Microsoft Windows Server 2003 o Microsoft Windows XP Professional.
- Procesador Pentium (se recomienda 1GHz o más).
- 256MB de RAM o más es recomendado.
- Tarjeta de red Ethernet.

4.4.2 Cliente

Desde el punto de vista del cliente, que es quien cumple los roles de generar y emitir o consumir el contenido deben considerarse ciertos requerimientos básicos en hardware y software:

- Microsoft Windows XP (SP2) o Microsoft Windows Vista.
- Microsoft .NET Framework v3.5.
- Microsoft Sql Server 2005 Compact Edition.
- DirectX v9 o superior.
- Windows Media Player v9.0 o superior.

La infraestructura con que debe contar el cliente que desee ejecutar la aplicación se presenta en el siguiente detalle:

- Procesador Intel 1GHz o superior.

- 1GB de memoria RAM o más.
- Tarjeta de red Ethernet.
- Conexión a Internet de 512/128Kbps (recomendado 1024/128Kbps).

En caso de emitir entrevistas en vivo se debe contar además con:

- Cámara web.
- Micrófono.
- Ambiente correctamente iluminado.

4.5 Solución tecnológica

A continuación se describirán las tecnologías elegidas para la realización del proyecto, estas fueron seleccionadas teniendo en cuenta su 'joven' incursión en el mercado, por considerarse novedosas y por su compatibilidad con el producto a desarrollar.

4.5.1 Windows Presentation Foundation (WPF)

Como es sabido, la interfaz de usuario cada día toma mayor importancia. En lo que respecta a nuestro producto, no fue la excepción. Se tomó esto en cuenta y se aplicaron en su desarrollo plantillas Windows Presentation Foundation (WPF) [ver anexo A2-3. Windows Presentation Foundation (WPF)].

La aplicación 'ofrece-requiere' un amplio diseño de interfaces. Se apunta a que el usuario se sienta 'atrapado' por ella, navegando y encontrando fácilmente todo lo que busca.

Desde el punto de vista de diseñador-desarrollador se vio la posibilidad de dejar la puerta abierta para explotar la interfaz de usuario en el futuro, es decir, hoy se presenta un producto, pero el día de mañana, sabemos que será posible hacer uso de esta plantilla con pocos ajustes y logrando muchas mejoras, dotando de mayor atractivo a la aplicación y por ende incrementando su valor comercial.

4.5.2 Windows Communication Foundation (WCF)

Windows Communication Foundation fue la tecnología .NET elegida ya que se encontró en ella todo lo que nuestra arquitectura de servicios necesitaba [ver anexo A2-4. Introducción a Windows Communication Foundation (WCF)].

La aplicación requería un Modelo de programación orientado a servicios y en WCF lo encontramos. Asegurándonos la interoperabilidad, así como también mensajería segura, fiable y organizada en transacciones.

4.6 Implementación

La implementación se ha desarrollado básicamente mediante el uso de servicios Web: Windows Communication Foundation (WCF) y herramientas de desarrollo Windows Presentation Foundation (WPF).

No queremos dejar de mencionar el uso de hilos, librerías específicas para la manipulación del streaming de video, FTP para el manejo de archivos, que si bien no aportaron a la solución adoptada se investigó sobre estas posibilidades.

De los tipos de servicios disponibles en WCF sólo dos se han utilizado en el proyecto. Los que funcionan como un servicio Web, *wsHttpBinding*, y los que siendo servicios Web utilizan métodos *callback* ideados para la doble comunicación, *wsHttpDualBinding*, es decir el servidor publica una interfaz que no sólo detalla los métodos a los que puede llamar el cliente sino los que el propio cliente ha de tener porque el servidor los va a invocar.

Ambos se han utilizado valiéndose de la infraestructura de Internet Information Services (IIS).

En cuanto a la forma de crear la aplicación mediante WPF, esto permitió dejar de lado antiguos paradigmas de creación de interfaces.

La interfaz se independiza del código, separando claramente la labor de diseño de la de desarrollo. Esta manera de trabajar está basada en el lenguaje XAML que deriva de XML por lo que la declaración de controles de usuario estará formada por etiquetas y atributos.

4.6.1 Implementación de acceso a datos

Como ya se explicó anteriormente, los datos en el servidor se mantienen en una base de datos creada en Microsoft Sql Server Express Edition. En esta base se mantienen las tablas necesarias para tener disponible información de las entidades relevantes del sistema, de relaciones entre ellas, y también algunos parámetros de configuración.

Para acceder a los datos de la base, se cuenta con componentes de la capa de acceso a datos. Estos componentes están constituidos por clases asociadas a las entidades de la base, llamadas clases entidades. A su vez, cada clase está definida por dos clases parciales. Una de ellas es generada automáticamente a partir de una herramienta de Visual Studio llamada Diseñador relacional de objetos. Este diseñador permite modelar las tablas de la base de datos con clases parciales del tipo Linq to Sql. Posteriormente, se crea para cada clase parcial generada de esta manera, otra clase parcial en donde se agregan las definiciones necesarias para concluir una única clase con la totalidad de la funcionalidad. Se implementa en C sharp y utiliza las APIs de Linq to Sql para trabajar en conjunto con las clases parciales generadas.

Para que los clientes puedan acceder a la información de la base de datos se implementan servicios Web que se publican en el servidor de aplicaciones. Estos servicios instancian objetos de las clases entidades para obtener la información de la base. En su implementación se definen estructuras de objetos a ser compartidos con los clientes. Algunas propiedades de estos objetos se asocian internamente a algunas propiedades de las instancias de los de las clases entidades. De esta manera se decide qué subconjunto de los datos que se manejan del lado del servidor pasarán al lado del cliente. Además, se agrega cierta lógica de negocios en los servicios, que se verán reflejadas en las propiedades de los objetos de las clases entidades. Los servicios se implementan como servicios de Windows Communication Foundation (WCF).

En forma análoga, los datos del lado del cliente se guardan en una base de datos de la misma familia que la que se encuentra del lado del servidor, este caso de Microsoft Sql Server Compact Edition. Es una base más ligera, no sólo por la versión de Sql Server usada, sino también por el volumen de información que maneja. De los datos que se encuentran en la base del servidor, un subconjunto pasará a formar parte de la base local. Básicamente las estructuras se mantienen, salvo por algunas tablas que contienen menos campos. Los datos que se mantienen son estrictamente los asociados a entrevistas que hayan sido descargadas por los usuarios. Estos datos se actualizan en cada descarga realizada.

De la misma manera que se cuenta con un componente de acceso a datos del lado del servidor, también se desarrolla uno del lado del cliente. Se cuenta con clases del Framework de .NET para ejecutar las consultas sobre la base de datos aplicando la sintaxis asociada. Prácticamente no se agrega lógica de negocios en este componente. Solamente se utiliza para guardar información que se obtiene del servidor en la base local y para consumir información a ser usada en la presentación cuando se trabaja sobre los datos locales.

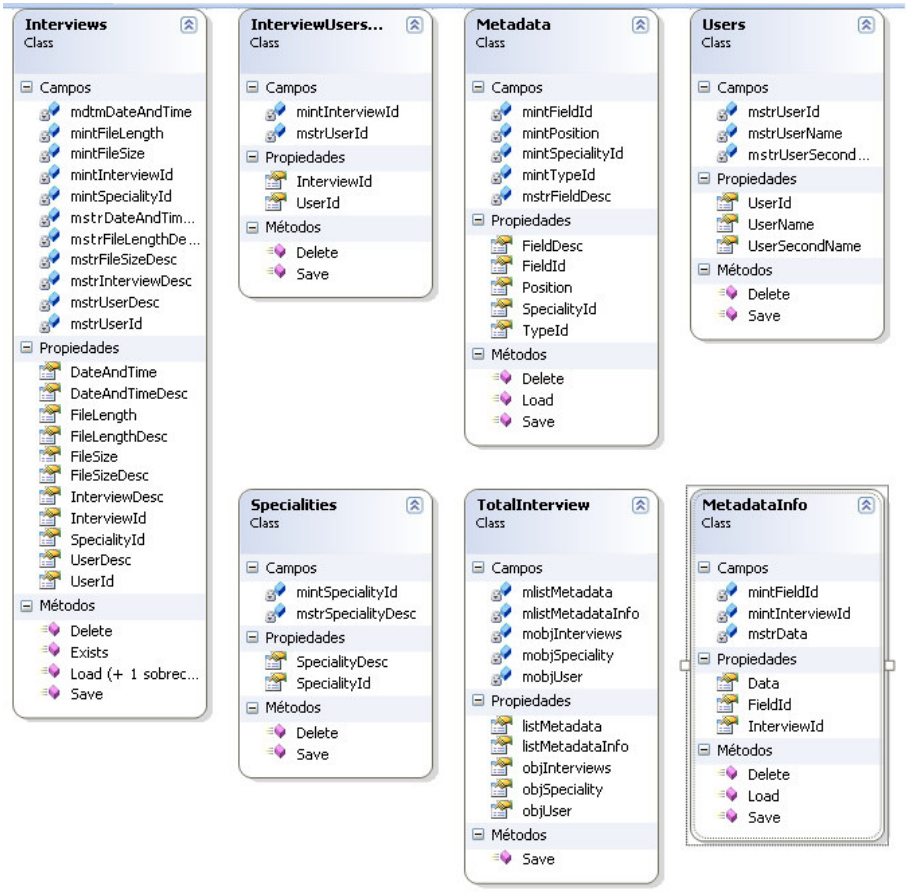


Figura B4.11: Diagrama de clases del componente local.

Es a partir de la interfaz de usuario que se relacionan los servicios Web que proporcionan la información desde el servidor con las clases que acceden a los datos locales. En algunos eventos que se disparan en la interfaz, cuando se ejecutan las funcionalidades asociadas, se puede ver como los datos de los servicios funcionan como entrada para la información local. A modo de ejemplo, cuando el usuario solicita, haciendo click en un botón de la interfaz, que se descargue una entrevista, se ejecutarán los métodos que cargan los objetos que acceden a la base local con las propiedades de los objetos devueltos por los servicios Web. Posteriormente, se ejecutará algún método que permita guardar las propiedades del objeto en la base.

Detalles de las estructuras de las bases de datos del servidor y del cliente y de los componentes que resuelven el acceso a datos en anexos [ver anexo A3-1. Base de Datos], [ver anexo A3-3. Componente local].

4.6.2 Desarrollo de la presentación

La presentación fue desarrollada a través de Windows Presentation Foundation (WPF). Se trata de desarrollar o dejar la puerta abierta a interfaces de usuario con un rico componente visual; con un fuerte componente de integración de diferentes funcionalidades como ser video, audio, documentos; con una gran cantidad de controles y propiedades que permiten visualizar y organizar diferentes tipos de información, aprovechando al máximo el área disponible.

Se aprovecha de WPF la arquitectura de Modelo–Vista–Controlador que implícitamente tiene relacionada, para poder separar la lógica de negocios de las interfaces de la aplicación. También se aprovecha la posibilidad de separar las funcionalidades propias de la programación

con lo que es el diseño de las interfaces, debido al lenguaje declarativo XAML usado para definir las interfaces.

La presentación está formada por una ventana ocupada mayoritariamente por un control que permite dividir las funcionalidades de la aplicación y que permite acceder a diferentes pestañas. A través de las pestañas se puede navegar entre un grupo de funcionalidades. Según el caso, se cuenta con un grupo de contenedores asociados a cada pestaña. En la presentación se utilizan varios tipos de contenedores. Dentro de cada contenedor, se tienen botones que permiten confirmar o cancelar las operaciones a realizar o navegar entre los contenedores incluidos en cada pestaña.

Es común el uso de controles básicos como ser etiquetas, cuadros de texto, cuadros desplegables, visualizadores de texto, contenedores de imágenes, barras de progreso, cajas de verificación, listas de elementos. También se usan controles que proporcionan funcionalidades más concretas o que permiten desarrollar una mejor interfaz.

En la pestaña de Publicaciones los usuarios tienen disponible una lista de enlaces de interés y un grupo de publicaciones de artículos. Estos enlaces y publicaciones están disponibles para todos los usuarios. Las funcionalidades que se encuentran aquí son: publicar un link que resulte de interés para otros usuarios; publicar artículos de tipo .doc o .pdf que quedarán disponibles para el resto de los usuarios; realizar búsquedas sobre artículos ya publicados; navegar sobre la lista de enlaces de interés o sobre los artículos.

Para presentar la información de esta sección se usa un control que permite visualizar las listas de enlaces y publicaciones como si pertenecieran a un documento. Este control permite paginar la información permitiendo el fácil desplazamiento y también permite realizar búsquedas de textos.

En la pestaña de Emisiones Grabadas se encuentran disponibles todas las entrevistas que ya se han realizado y que se encuentran disponibles para ser descargadas. Las entrevistas disponibles para cada usuario serán aquellas a las que fue invitado alguna vez y que estén asociadas a la especialidad seleccionada por el usuario. Puede verse la lista de entrevistas con información relevante como ser una descripción, el usuario que la publicó, el tamaño y largo del archivo, la cantidad de descargas que ha tenido y un valor asociado al ranking, además de una vista previa de la entrevista. El usuario podrá acceder a cualquiera de las entrevistas, ver la información completa de la misma y descargarla. En esta sección aparte de los controles comunes se utiliza un contenedor que permite que sus elementos se apilen de izquierda a derecha realizando un corte cuando se ha ocupado el ancho máximo y haciendo un corte de la información hacia abajo. También proporciona flechas de desplazamiento para alcanzar la información no visible. En este control es en donde se ubican las imágenes de vista previa de las entrevistas.

En la pestaña de Emisiones On-line se cuenta con una lista de las entrevistas a las que el usuario ha sido invitado a presenciar. Una vez seleccionada la entrevista de interés podrá acceder a la emisión en vivo de la misma. Se ve aquí el uso de listas y grillas, así como también controles que permiten manejar la reproducción de contenido multimedia.

En la pestaña de Emitir se cuenta con una serie de botones que permiten navegar hacia otros paneles en donde se puede realizar diferentes tareas sobre las entrevistas: iniciar una entrevista, definir los datos de la portada, publicar su emisión, emitirla en vivo, realizar el cierre de la entrevista o publicarla. Se utilizan en esta sección muchos controles comunes combinados de manera diferente, según la necesidad. Para la parte de emisión en vivo se utilizan controles especiales para permitir la captura y reproducción del contenido multimedia.

En la pestaña de Descargas se tienen una lista con las entrevistas que ya han sido descargadas localmente, junto con información adicional. El estilo de esta interfaz es similar al de la pestaña Emisiones Grabadas.

En la pestaña de Administración se permite dar de alta o modificar información de administración de especialidades, usuarios y metadata. Para esto se usan interfaces de fácil uso, a través de controles comunes. Su navegabilidad se resuelve con botones que indican de forma clara al usuario la funcionalidad que proveen.

WPF permite automatizar la asignación de valores a propiedades de objetos a través de *data binding*. Este concepto implica que se puede asociar un objeto o algunas de sus propiedades a controles o propiedades de ellos. Se utiliza *binding* entre objetos y controles de la interfaz en la mayoría de ellas. También se utiliza *binding* entre controles a nivel de interfaz, para contar con una interfaz más automatizada. Un ejemplo puede ser cuando se asocian los anchos de varios controles. El cambio de uno de ellos implica el cambio de todos en forma automática.



Figura B4.12: Presentación de G-Sell.

También se usa en la presentación definiciones de estilo. De esta manera puede definirse un estilo a ser usado por varios controles, como un conjunto de valores para propiedades de objetos del mismo tipo. A modo de ejemplo, se puede indicar en la definición de un estilo el tipo de fuente, estilo, tamaño, color, color de fondo, ancho, alto de un conjunto de etiquetas.

4.6.3 Desarrollo de captura y streaming de multimedia

La base para el desarrollo de aplicaciones Windows multimedia es construida por medio del uso de las distintas componentes que integran Direct [ver anexo A4-2. Multimedia en Windows]. En particular, DirectShow y DirectSound ofrecen a las aplicaciones las funciones necesarias para implementar la captura de audio y video. A través de DirectShow es posible construir un grafo de filtros que modele y controle el comportamiento de dicha captura.

Para resolver la grabación a disco y emisión en vivo Microsoft presenta el paquete Windows Media SDK incluyendo componentes que cuentan, entre otras, con la capacidad de definir y manipular el formato de archivo, así como el envío del flujo generado a través de la red.

La librería WindowsMediaLib.NET (versión de este paquete para .NET) proporciona el marco adecuado para incorporar dichas funcionalidades en la solución final.

De forma de estandarizar se opta por definir que el formato para los filtros render (ASFWriter) encargados de grabar y emitir sean cargados a partir de una identificación única (GUID) obtenida desde un archivo de configuración disponible en todas las instalaciones de Windows.

Como se explicó en Diseño, por motivos de calidad en el video se decidió separar el resultado de la captura en dos flujos con calidades y formatos distintos. De forma de minimizar el impacto en consumo de recursos que esto implica, se usa un filtro de transformación que copia la entrada recibida a varias salidas independientes, permitiendo así emitir al servidor de streaming (con baja calidad aunque aceptable) y grabar el archivo en forma local con otra calidad mucho más alta.

Es así que el grafo de filtros generado para modelar la emisión de una entrevista en vivo consta de filtros source que permiten la captura directamente desde los dispositivos, filtros de transformación que cumplen la labor antes mencionada, y filtros render que entregan el flujo multimedia capturado a los dispositivos locales o remotos correspondientes. Estos filtros son conectados de determinada forma para hacer fluir la información a través del grafo desde la fuente hasta los distintos elementos de entrega.

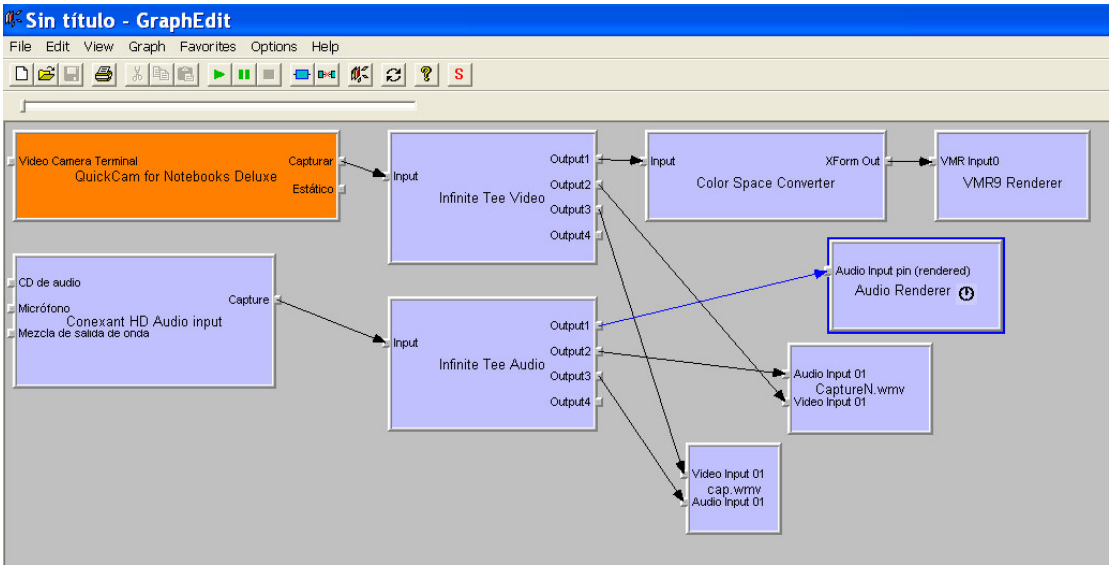


Figura B4.13: Grafo de captura y emisión.

Los identificadores únicos son strings que se usan, en este caso, para configurar el filtro definiendo los parámetros del formato ASF que tendrá cada flujo. Más específicamente, los archivos multimedia grabados para reproducción local se definen con formato Windows media video 8 (WMV8), recomendado para contenidos de audio/video de alta calidad (1400Kbps, 30fps, 320x240). Para las audiencias con anchos de banda de 28.8 a 100Kbps, se recomienda configurar un perfil de flujo de calidad más baja (30fps 176x144) para la transmisión vía red.

La representación gráfica del nivel de audio capturado es implementada por medio de muestreos consecutivos obtenidos desde un buffer de captura que mantiene y usa la librería DirectSound.

En la presentación al usuario de la captura se utiliza una versión modificada de MediaKit que permite realizar todas las funciones necesarias para capturar y enviar multimedia a través de WPF. Esta librería empaqueta un conjunto de controles personalizados que brindan funcionalidad DirectShow a las aplicaciones WPF.

Para resolver la necesidad planteada, agregando audio y streaming, se agregó la clase CaptureElement al paquete original y se ajustaron aquellas que estaban relacionadas [ver anexo A4-1. Captura y streaming multimedia].

En la etapa de generación de la emisión en vivo se debió implementar un conjunto de métodos que permitieran establecer diálogo con el servidor de streaming con el fin que éste le informe a la aplicación cliente de eventos acontecidos.

Esta clase es además la encargada de proporcionar las credenciales necesarias para que la aplicación emisora pueda crear un punto de publicación en el servidor y así distribuir el contenido a la audiencia.

4.6.4 Desarrollo de chat

Se trata de la creación de un sistema participativo en la que tanto el emisor como los invitados a presenciar la transmisión pueden participar, durante y después de ella. Se podrá

ver la lista de nombres de usuarios conectados al chat y por lo tanto identificar a aquellos que están asistiendo a la entrevista.

En el momento que el usuario intenta acceder a una emisión en vivo, automáticamente se conecta al chat de la misma, pudiendo hacer uso de este aunque la transmisión no haya comenzado. Las preguntas y respuestas que se van generando durante el transcurso de la emisión podrán ser vistas tanto por el emisor como por los consumidores de ella.

Solamente los usuarios invitados a presenciar una entrevista podrán acceder a ellas, pasando el control previo de autenticación en la base de datos.

El componente de chat se implementó por medio de la utilización de servicios Web básicos (wsHttpBinding) que permiten lograr la independencia del entorno y generalidad buscada. Se resolvió el intercambio de mensajes durante la emisión como una cola de mensajes en donde cada mensaje enviado por el usuario es guardado en el servidor para ser consumido por los demás usuarios conectados al chat.

4.6.5 Desarrollo de publicación y descarga

La aplicación cliente necesita contar con la posibilidad de transferir archivos al servidor en ambas direcciones. Estas transferencias de subida (publicación) y bajada (descarga) son efectuadas a través de un servicio Web WCF alojado en un servidor de aplicaciones (IIS v6) en el extremo servidor.

El servicio usa como modelo de comunicación basicHttpBinding que como su nombre lo indica envía y recibe los mensajes desde la aplicación mediante el protocolo HTTP. Al tratarse de un servicio que se encarga de la transferencia de archivos, y que los mensajes pueden tener tamaños considerables, se ajustaron ciertos parámetros en su configuración. Por lo tanto, se incrementaron en la aplicación cliente los timeouts de recepción (diez minutos) y envío de mensajes (cinco minutos), así como el tamaño máximo por mensaje recibido (10067108864 Bytes).

Una entrevista realizada consta de tres archivos (audio/video, chat y portada) que para ser subidos al servidor deben ser empaquetados. Valiéndose de la librería DotNetZip, versión de Zip para .NET, se forma un archivo con formato y extensión zip encriptado con contraseña.

Aprovechando que un altísimo porcentaje de la información a empaquetar se trata de video, se genera el archivo formato zip sin compresión para no aumentar la demora experimentada por el usuario al publicar la entrevista.

Tanto las tareas de publicación, como las de descarga están contenidas en un hilo independiente al que ejecuta la aplicación, por lo que se adopta cierta simultaneidad de tareas que permite al usuario descargar una entrevista mientras realiza otras operaciones [ver anexo A2-5. Publicación y descarga de archivos].

Publicación

En el momento que el cliente intenta publicar una entrevista en el servidor, la aplicación genera el archivo zip usando el algoritmo de encriptación PKZip en la carpeta local predeterminada. A continuación esta información empaquetada es subida por medio del servicio Web a la carpeta destino definida por el sistema.

Una vez concluida esta acción, a través del mismo servicio se extrae del paquete la imagen de portada de la entrevista dejando toda la información necesaria disponible para luego ser descargada a demanda por los demás usuarios autorizados.

Descarga

La descarga de entrevistas publicadas en el servidor es iniciada desde la aplicación cliente. Esta inicia una transferencia a través del servicio Web que descarga el archivo zip desde el servidor y lo almacena temporalmente en la carpeta predeterminada.

Por último, lo desempaqueta obteniendo los tres archivos correspondientes a la entrevista bajada listos para ser consumidos. Como forma de mantenimiento y para evitar el

uso innecesario de disco se elimina el archivo extensión zip del equipo local una vez concluida la descarga.

4.6.6 Desarrollo de funcionalidades complementarias

Tanto para el consumo a demanda, como para la reproducción de entrevistas en vivo se utiliza el control MediaElement incorporado en forma estándar por WPF.

Este control ofrece versatilidad en el manejo y presentación de los elementos multimedia insertos en la ventana que lo contiene. En el caso de entrevistas que son vistas una vez descargadas, el progreso del video reproducido debe estar sincronizado con el intercambio de mensajes que se ha realizado en dicha entrevista. Esta sincronización se resuelve implementando distintos hilos de ejecución coordinados entre sí.

Librería Utilidades

Está compuesta por dos clases simples, pero muy usadas en toda la aplicación.

- Transformación: encripta con 3DES el número de entrevista a un string. Usada para identificar las entrevistas en un formato de nombre no reconocible.
- Chequeo de conexión: mediante un intento de conexión TCP verifica la existencia de conectividad

Representación de captura de audio

Se implementaron métodos que usan la librería DirectSound y manejo de hilos para resolver el las representación gráfica de la captura de sonido por medio de un medidor de volumen.

Clase ImageButton

Extiende el control *Button* estándar de WPF. Es un control usado para definir botones especiales personalizados.

4.7 Dificultades

En la segunda etapa del trabajo, una vez analizados los requerimientos y haciendo foco en el diseño de la solución que modelaría la *Cámara Gesell* virtual surgieron obstáculos destacables por su incidencia final.

En el momento de incursionar en la captura y streaming de contenido multimedia se debió lidiar con una de las dificultades vinculadas al ancho de banda de subida en las actuales conexiones a Internet disponibles en el mercado [ver anexo A4-4. Servidor de streaming]. Esta barrera tecnológica, sumado a que no existen al momento en DirectShow, filtros estándares que permitan enviar flujo por la red usando los protocolos óptimos para streaming, condicionan en alto grado las transmisiones en vivo. Esto no permite la posibilidad de trabajar con buenas calidades de emisión, llegando al punto de experimentar interrupciones en la continuidad del flujo por repetidos buffering en emisiones de calidad media.

Otro inconveniente relacionado con el tratamiento del contenido multimedia fue la incapacidad de definir en forma estándar perfiles de formato WMV9, no pudiendo de esta forma sacar el máximo provecho a los avances que esta nueva versión del formato aporta.

Al modelar el subsistema de intercambio de mensajes en una entrevista se comenzó con el estudio de distintas aplicaciones servidores de chat existentes que podrían dotar al sistema de la funcionalidad buscada. Con este fin se investigaron, entre otros, Openfire y Ejabberd con sus respectivos clientes Spark y Psi como posibles candidatos a formar parte de la aplicación final. Al tratarse de códigos cerrados y ante la incapacidad de adaptar e integrar tales soluciones con el producto en desarrollo no fue posible con estas pruebas lograr que emisión multimedia y chat trabajaran en conjunto de forma transparente para el usuario.

4.8 Decisiones tomadas

En lo que refiere al tratamiento de contenido multimedia se decidió utilizar DirectShow en su versión para .NET. Con el fin de resolver algunos inconvenientes que esta librería presenta al integrase con WPF, se resolvió adaptar el funcionamiento del paquete MediaKit estándar a la medida de los requerimientos de captura de audio y streaming.

La librería DirectShow.NET no incluye en forma estándar filtros que permitan transportar el streaming de contenido a través de la Internet con UDP, por lo que se decidió trabajar con archivos en formato ASF transmitidos vía TCP (HTTP).

En este caso -y para sobrellevar la fuerte restricción de ancho de banda encontrada- se decidió dividir la captura en dos flujos de datos independientes que permitieran grabar en un formato óptimo para su reproducción posterior y enviar a través de la red en una calidad aceptable para ser visto sin interrupciones por la audiencia invitada.

La definición del formato de grabación a disco para la captura multimedia se realizó por medio de WindowsMediaLib.NET (versión .NET del SDK de Windows Media). Debido a la incapacidad de definir perfiles WMV9 cargados vía GUID, se optó por manejar el formato de la versión anterior (WMV8).

Por último, se tomó la decisión de implementar una versión propia de chat trabajando con servicios Web y manejo de archivos con formato XML para modelar los requerimientos básicos identificados.

Para mantener información del lado del cliente se decidió no usar una base de datos explícita. No sólo porque debería ser una base de datos sin costos para los usuarios, sino también porque generaría problemas visibles al usuario a la hora de instalar el sistema. Por tal motivo se optó por usar una base de datos embebida en el sistema, la base creada en Sql Server Compact Edition. De esta manera la base de datos y los componentes necesarios para su funcionamiento se instalarán de forma no visible al usuario.

5. Pruebas y resultados

5.1 Pruebas

Las pruebas del producto *G-Sell* fueron realizadas como testeo del producto generado, en base a un conjunto de datos de prueba incluidos a modo de ejemplo de uso del sistema. La naturaleza de los componentes identificados en la solución final permitió realizar en un principio pruebas unitarias, verificando de esa forma funcionalidades particulares y luego pruebas de integración concluyendo así el testing de toda la aplicación. Se detallan a continuación las pruebas realizadas en cada módulo.

5.1.1 Pruebas realizadas al servicio de chat

En este punto debemos recordar que:

- Para comunicarse entre sí los usuarios deben compartir la misma entrevista que se está emitiendo. No pueden comunicarse usuarios que estén viendo entrevistas diferentes en un mismo momento.
- La implementación del chat se desarrolló de dos maneras diferentes: por servicios dúplex y por servicios Web. En la versión final del producto se usaron servicios Web para implementar la cola de mensajes.

Se detalla la lista de casos de uso planteados para su testeo. Su ejecución se realizó en forma incremental en el orden que se expone a continuación.

En la implementación desarrollada por servicios dúplex se verificó [ver anexo A2-7. Servicios dúplex]:

1- La comunicación entre dos usuarios conectados en la misma red local, sin tener en cuenta el control de pertenecer ambos a la misma entrevista.

2- Se incorporó el control anteriormente descripto. Es decir, que puedan chatear siempre y cuando estén conectados a la emisión de una misma entrevista.

3- La comunicación entre dos usuarios con conexión a Internet mediante direcciones IP públicas.

4- Se extendió el caso de prueba a cuatro usuarios concurrentes en la sala de chat, dos pertenecientes a una misma entrevista y dos a otra diferente. Con este caso se verificó que usuarios pertenecientes a entrevistas diferentes no pudieran chatear entre sí y que no se vieran en la interfaz de usuario.

5- Se verificó que cada vez que un usuario ingrese o se retire de la sala de chat, se actualice la lista de usuarios conectados a la entrevista.

Hasta este punto las pruebas resultaron satisfactorias. Sin embargo, cuando se intentó extender las pruebas al caso de comunicación entre usuarios cuyas direcciones IP podían o no ser públicas, eventualmente detrás de routers, se presentaron inconvenientes.

Ante la imposibilidad de aplicar este tipo de servicios para modelar el servicio de chat, se implementó una alternativa basada en servicios Web.

En la implementación usando servicios Web se resuelve el intercambio de mensajes entre usuarios que comparten una entrevista como cola de mensajes. En este contexto las pruebas realizadas fueron:

1- Que cada vez que un usuario ingresa a la emisión de una entrevista, quede registrado en la tabla asociada de la base de datos.

2- Que cada vez que un usuario abandona la emisión de una entrevista, se elimine el registro de la tabla.

3- Que la lista de usuarios que ingresaron a la emisión de determinada entrevista y aún no la abandonaron se mantenga actualizada en las interfases de todos los usuarios conectados a la misma entrevista.

4- Que los mensajes enviados por los usuarios queden registrados en la tabla de mensajes de la base de datos, en un estilo cola de mensajes.

5- Que los mensajes puestos en cola de mensajes sean procesados en el orden correcto, sin repetición de mensajes ni falta de procesamiento de ninguno de ellos.

5.1.2 Pruebas realizadas al consumo por demanda

Las pruebas en este módulo también se ejecutaron en forma incremental, se comenzó con el consumo de un video de prueba bajado de Internet invocándolo desde Windows Media Player. Como se ha descripto anteriormente en una primera instancia se intentó adaptar esta herramienta a *G-Sell* descartándola, ya que el consumo a demanda requería mucho más que la ejecución de un video local.

Se procedió asi a desarrollar esta funcionalidad (las que ofrece Windows Media Player) mediante las implementaciones descriptas en la sección 5.1.1.Finalmente se testea la ejecución del video (video de prueba bajado de Internet) desde la aplicación (*G-Sell*).

Concluida esta etapa se continuó con el consumo de un video, pero esta vez generado por la propia aplicación *G-Sell*. Se usó un video que es el resultado de una entrevista en vivo grabado en el disco local. Se verificó cada una de las funcionalidades desarrolladas: ejecutar, pausar y detener.

Luego de que el video generado por *G-Sell* fuera consumido a demanda y pasando satisfactoriamente las pruebas realizadas, se pasa a una segunda etapa que fue la incorporación del chat. Esto se realizó secuencialmente. Primero se comenzó con el testing de manejo de hilos y procesos concurrentes. Aquí se testeó la ejecución del video y el chat concurrentemente y transcurrida esta etapa se pasó a testear su sincronización para constatar que los mensajes escritos en determinado momento de la entrevista en vivo se desplegaran de igual manera cuando su consumo se hiciera a demanda.

5.1.3 Pruebas realizadas sobre video

De aquí en adelante se utilizará el término 'codificador' para hacer referencia al usuario que inicia la transmisión en vivo y 'receptor' para el usuario que la consume.

Se probó la captura y emisión de contenido en forma local, esto es, el servidor de streaming -instalado en una máquina virtual-, el cliente codificador (encoder) y un cliente receptor dispuestos en la misma red LAN.

A continuación se realizaron pruebas en un ambiente más ajustado a la realidad planteada, donde la emisión se transmitiera a través de Internet. Por medio de estas primeras pruebas de multimedia en Internet fue posible detectar degradaciones importantes en la fluidez de la emisión. Estas dificultades llevaron a experimentar con varias resoluciones en la emisión hasta llegar a una calidad aceptable, pero baja. Esto motivó que finalmente se decidiera grabar la emisión en el archivo local en alta calidad y emitir en vivo en una calidad inferior.

La calidad de imagen capturada depende en gran parte del dispositivo con el cual se captura. Se realizaron varias pruebas con cámaras web de distintas resoluciones e incluso algunas con micrófono incorporado, que permitieron validar la aceptación de un amplio espectro de estos dispositivos de captura.

Una vez establecidos los formatos multimedia adecuados se realizaron pruebas de integración entre el chat y la emisión en vivo, tanto captura, como emisión y recepción, probando de esa manera que sólo usuarios habilitados a cierta entrevista en curso pudieran ver dicha transmisión y escribir mensajes a ser vistos por todos los usuarios invitados y presentes.

5.1.4 Evaluación de performance para emisión

Se han desarrollado pruebas en varios escenarios posibles obteniendo resultados tan diversos como las propias pruebas realizadas. A continuación se detalla el conjunto de pruebas llevadas a cabo categorizando según el medio usado.

Transmisión sobre LAN

Se considera importante destacar que para todos los ensayos realizados en una red LAN (100Mbps o 1Gbps) como soporte, los resultados obtenidos son óptimos, cualquiera sea el formato multimedia elegido.

En sucesivas pruebas cumplidas se han logrado grabar y emitir a través del servidor de streaming entrevistas de una, dos, tres y hasta cuatro horas de duración sin inconvenientes.

Transmisión sobre Internet

Todas las pruebas evaluando la performance sobre Internet mencionadas en este análisis fueron realizadas con líneas ADSL de diferentes capacidades: 512/256kbps, 1024/ 128Kbps y un enlace empresarial proveído por *EviMed*.

Por razones expuestas en la sección 4.3 Diseño, la transmisión del flujo multimedia desde el cliente codificador al servidor de streaming debe realizarse sobre TCP, vía el protocolo HTTP.

Considerando la restricción en cuanto a formato de archivos soportados por el servidor de streaming (WMS) y la profunda flexibilidad que brinda el entorno de desarrollo para soportar Windows Media Format (WMF) se decidió probar en una primera instancia con el formato ASF.

Inicialmente se intentó distribuir el contenido sobre TCP (HTTP) archivos comprimidos con formato WMV9 sin tener los resultados esperados, por lo que se pasó a definir y experimentar con diferentes perfiles de codificación WMV8 transmitidos en TCP y UDP.

A continuación se detalla un conjunto de pruebas realizadas intercambiando los distintos actores de la emisión en vivo, combinando los enlaces disponibles para generar diferentes escenarios.

- Enlace AntelData Empresarial – Enlace AntelData Home.
  En este caso se colocó el servidor de streaming (WMS) en el enlace empresarial y se realizaron pruebas con distintas combinaciones de ADSL en las aplicaciones codificador (encoder) y reproductor. En todos los experimentos se llegó a emitir más de dos horas sin problemas.
- Enlace AntelData Empresarial – Enlace Dedicado.
  Se dispuso la misma configuración que en el caso anterior, pero esta vez se conectaron los clientes reproductor y codificador a través del enlace Dedicado. El resultado fue exitoso en todas las pruebas realizadas.
- Enlace Dedicado.
  En esta prueba se dispuso de la aplicación encoder, el servidor de streaming WMS9 (corriendo en una máquina virtual) y la aplicación de reproducción ejecutándose en una única máquina física obteniendo dos direcciones IP públicas de Dedicado. Los resultados obtenidos fueron poco convincentes, detectando un alto uso de CPU y memoria en el equipo, posiblemente no asociado a problemas de ancho de banda.
- Enlace Dedicado – Enlace AntelData Home.
  Para esta combinación de enlaces se realizaron dos pruebas. En ambas se usó el servidor de streaming WMS presentado en el enlace Dedicado y la aplicación reproductor (player) ejecutada en los dos enlaces indistintamente. El cliente

emisor, que es quien codifica la señal, fue colocado en cada uno de los enlaces involucrados.

- Codificador en enlace Dedicado
  Se probó la aplicación encoder colocada en Dedicado, puesta en una IP distinta a la del servidor simulando de esta forma el cliente ADSL obteniendo resultados muy pobres.
- Codificador en enlace AntelData Home
  La aplicación encoder se conecta desde el enlace de AntelData al servidor alojado en Dedicado obteniendo un mejor resultado (probablemente menos carga de CPU) pero igual inaceptable para una entrevista en vivo.

En ambos casos se intentó usar UDP aprovechando la facilidad del protocolo MMS rollover de Microsoft, observando una leve mejoría, pero aún manteniendo el resultado por debajo de niveles aceptables.

Tomando en cuenta que el envío de flujo de datos desde el cliente codificador al servidor de streaming se está realizando a través de Internet sobre TCP, se intentó simplificar el esquema distribuyendo únicamente un archivo con formato WMV almacenado localmente en el servidor. Como resultado de esta prueba se puede apreciar una importante degradación de la emisión captada. De esta forma, es posible descartar el supuesto de que los inconvenientes encontrados sean originados en la subida de los datos desde el generador.

5.1.5 Pruebas sobre acceso a datos

Para cada entidad representada en la base de datos, se asocia una clase para acceso a datos. En las pruebas unitarias realizadas se testea para cada clase todos los métodos implementados. Básicamente, alta, baja y modificación de registros de la base. Queda implícito el testeo de las asignaciones de propiedades. Esto se observa cada vez que se testea alguno de los métodos mencionados. Estas pruebas se realizan sobre una base de datos local.

Para cada clase de acceso a datos, se tiene un servicio asociado. Para testear el funcionamiento de los servicios que exponen la información desde el servidor, luego de implementar la base *Gesell* en el servidor de base de datos y de exponer los servicios, se invocan sus métodos desde un equipo cliente. Se prueban los métodos ofrecidos por los servicios. Mayoritariamente los servicios expuestos realizan operaciones de alta, baja y modificación de datos de la base. En algunos casos es necesario también testear la incorporación de lógica de negocios aplicada en los métodos de los servicios.

Al realizar las pruebas de los servicios también se testean los permisos necesarios en el servidor de base de datos para poder ejecutar las consultas y los permisos en el servidor de archivos para acceder a los archivos de datos de la aplicación, la correcta ubicación de los archivos y publicación de los servicios que deben estar en el servidor de aplicaciones. Debe tenerse en cuenta que en las pruebas del sistema, el servidor de aplicaciones y el de base de datos concluyen en un mismo servidor, el de la máquina virtual.

5.1.6 Pruebas sobre la presentación

En la capa de presentación se realizan diferentes pruebas sobre el funcionamiento de los diferentes controles incluidos en las interfaces. Algunos controles usados son bastante simples pero algunos otros son un poco más complejos. Se prueba la navegación en el control principal, que provee las pestañas, que ocupa la mayor parte de la ventana, así como de los paneles que se incluyen en cada una de sus pestañas. En cada interfaz se están testeando siempre los accesos a datos, ya que en todas partes se tiene información de la base existente en el servidor o de la base de datos local. Principalmente, la presentación es usada como testeo de integridad del sistema. Las funcionalidades que fueran testeadas en forma unitaria anteriormente, pasan a funcionar en conjunto.

En la pestaña de publicaciones se hacen pruebas de:

- Acceso a datos de publicaciones.
- Descarga de documentos desde el servidor de archivos.
- En forma simultánea, alta de información en la base de datos y subida de archivos al servidor de archivos.
- Navegabilidad.

En la pestaña de emisiones grabadas y disponibles para ser descargadas se hacen pruebas de:

- Conjuntamente: acceso a datos de las entrevistas del servidor de base de datos, filtros aplicados a los datos que se visualizan, acceso a archivos de imágenes disponibles en el servidor de archivos. Se debe recordar que las entrevistas visibles son las que el usuario ha sido invitado a presenciar, que están asociadas a la especialidad seleccionada en la interfaz y cuyo estado es finalizada. También es importante que en la interfaz se visualiza una imagen de vista previa de la entrevista.
- Conjuntamente: descarga de archivos desde el servidor de archivos, obtención de datos desde el servidor de base de datos, modificación de los datos de la base de datos local. Cuando se descarga una entrevista se obtiene un archivo con la información del video, el chat y la imagen de vista previa. Se obtiene información de la base de datos del servidor y se replica en forma parcial en la base de datos local.
- Navegabilidad.

En la pestaña de emisiones a ser vistas en vivo se hacen pruebas de:

- Acceso a datos de las entrevistas del servidor de base de datos y la aplicación de filtros aplicados a los datos que se visualizan. La información que se visualiza corresponde a las entrevistas a las que fue invitado el usuario, que se encuentran en estado para ser emitidas y cuya fecha de emisión es posterior a la actual.
- Conjuntamente: acceso al streaming de video que se está realizando en vivo y visualización y envío de mensajes de los usuarios que acceden concurrentemente a la emisión de la entrevista.
- Navegabilidad.

En la pestaña de emitir entrevistas se hacen pruebas de:

- Acceso a datos de las entrevistas del servidor de base de datos y la aplicación de filtros aplicados a los datos que se visualizan. Las entrevistas deben ser filtradas por el usuario responsable de la misma y por el estado de la entrevista, que no deben ser finalizadas.
- Acceso a datos particulares de una entrevista y modificación de los mismos. Se permite modificar la portada, el cierre de las entrevistas, el momento de la emisión y los usuarios invitados.
- Conjuntamente: emisión del streaming de video, visualización y envío de mensajes de los usuarios que acceden concurrentemente a la emisión de la entrevista, grabación de los archivos locales con información de video, chat e imagen previa de la entrevista.
- Conjuntamente: modificación de datos de la base en el servidor, formación de un único archivo zip que contiene los archivos de video, chat e imagen previa, envió del archivo zip al servidor de aplicaciones. Estas funcionalidades se llevan a cabo en el marco de la publicación de la entrevista para que pueda empezar a ser descargada por usuarios con permisos suficientes.
- Navegabilidad.

En la pestaña de descargas de entrevistas se hacen pruebas de:

- Acceso a datos de las entrevistas en la base de datos local y la aplicación de filtros aplicados a los datos que se visualizan. Las entrevistas deben ser filtradas por el usuario que las haya descargado.
- Conjuntamente, reproducción del archivo de video y visualización de los mensajes intercambiados durante la entrevista.
- Navegabilidad.

En la pestaña de administración del sistema se hacen pruebas de:

- Alta, baja y modificación de información en la base de datos del servidor, en forma puntual.
- Navegabilidad.

Además de las pruebas de funcionalidad, se revisa la usabilidad del sistema. El mismo debe motivar al usuario en su uso. Se observa la aplicación de algunos de los patrones básicos para el desarrollo de aplicaciones con tendencia Web 2.0 [ver anexo A2-1. Ocho patrones básicos para el desarrollo de la Web 2.0].

Se valida el aprovechamiento de la inteligencia colectiva mediante:

- La participación de varios usuarios en una misma entrevista y la posibilidad de compartir mensajes y comentarios entre ellos durante la emisión.
- El poder invitar a presenciar la entrevista a un gran número de usuarios invitados.
- La funcionalidad que permite descargar entrevistas realizadas anteriormente, para ser consumidas en forma local.
- La posibilidad de dar de alta nuevos usuarios.
- La posibilidad de compartir publicaciones y enlaces de interés.
- Las búsquedas sobre la información registrada en el sistema.

Se evalúa la calidad que los datos puedan tener sabiendo que no es un sistema de uso abierto, sino que los usuarios son un grupo seleccionado. Según la industria en la que se utilice el producto podrá tener usuarios diversos y con un conocimiento suficiente para generar contenido de calidad.

Dado que la aplicación funciona usando Internet como base, se aseguran funcionalidades asociadas a las propias opiniones de los usuarios. En este ámbito se prueba el acceso a la información de votaciones sobre las entrevistas. También se testea la posibilidad de definir metadata en el sistema.

5.2 Resultados

Finalmente, después de realizadadas todas las pruebas parciales antes mencionadas, integrado cada módulo en el proyecto '*Cámara de Gesell*' virtual, y de testeada la integración de los mismos, se obtiene como resultado el objetivo planteado; una aplicación que permitirá a los médicos de la Asociación de Psiquiatras del Interior participar en forma directa de consultas en vivo realizadas por sus colegas, sin importar las ubicaciones físicas de los participantes. Y de la mano con esto, la posibilidad de generar conocimiento colectivo y de poder compartirlo en las diferentes maneras en las que se ha comentado anteriormente.

Y como una extensión muy importante a este objetivo, se logró una aplicación que puede ser usada en cualquier otro ámbito, no sólo en un entorno médico. Esto se debe a la libertad de poder configurar los metadatos asociados a las entrevistas, estableciendo así el tipo de información que contendrá el sistema.

El resultado no ha sido sólo el desarrollo de una aplicación para un sector de la medicina. Ha ido un poco más lejos llegando al desarrollo de un sistema aplicable a cualquier ambiente imaginable y capaz de manejar cualquier tipo de información.

6. Conclusiones y trabajo futuro

Se detalla en el presente capítulo las conclusiones a las que se ha llegado en las aplicaciones desarrolladas, así como también una perspectiva de continuidad en sus desarrollos.

6.1 Conclusiones

Efectivamente se alcanzó las metas planteadas y los resultados obtenidos son satisfactorios.

El objetivo inicial del proyecto fue generar nuevos ámbitos para fomentar la educación a través de Internet. Aprovechando la red social de *eviDoctor* ya existente, se pretendía proveer de nuevas herramientas que la aproximaran a las tendencias actuales de Web 2.0.

En tal sentido, se partió de un análisis exhaustivo de herramientas que permitieran alcanzar el objetivo planteado. Esto concluyó con la determinación de implantar *WikiMed*. Luego se pretendió integrarla a un sistema de gestión de cursos que permitiera crear comunidades de aprendizaje en línea. El estado del arte realizado permitió identificar una implementación avanzada en versión estable y de libre distribución de la integración de *MediaWiki* con *Moodle* (sistema de gestión de contenidos), por lo que se decidió interrumpir el desarrollo de una aplicación que la solucionara. Como resultado de esta etapa se puso en producción *WikiMed* con los resultados antes descritos, permitiendo a la empresa *EviMed* contar con un sitio donde la comunidad de médicos de *eviDoctor* puede compartir conocimiento en forma sencilla y rápida.

La segunda etapa del trabajo surgió ante la inquietud de un grupo de médicos psiquiatras del Uruguay, en el sentido de poder compartir las experiencias con sus pacientes, en vivo y en directo con otros colegas. Se trata de una rama de la medicina en que la 'observación' al paciente resulta crítico para un buen diagnóstico y tratamiento.

Para que esto fuera posible se desarrolló una *Cámara de Gesell* virtual (*G-Sell*), aplicación del estilo cliente inteligente (Smart Client). Cuando el sistema funciona de forma conectada, se ejecuta a través de Internet, salvando así la primer y más importante limitación que sufren estos médicos hoy en día, encontrarse físicamente en lugares diferentes. *G-Sell* les provee de transmisión en vivo (video y audio) y en directo (tiempo real) de la entrevista con los pacientes. Además permite enriquecer las entrevistas con el intercambio de mensajes entre los médicos a través del servicio de chat.

Se describe a continuación varios puntos a destacar los cuales cambiarían por completo la situación actual de trabajo de los médicos de la Asociación de Psiquiatras del Interior:

1- Oportunidad de evaluación al paciente para todos los médicos que participan mediada por el médico tratante. Debido a que todos presencian la entrevista en el mismo momento, el médico tratante quien se encuentra frente al paciente puede enriquecer la entrevista con los aportes de sus colegas.

2- Discusión del caso entre todos en tiempo real y con el paciente a la vista. El servicio de chat incluido en la aplicación permite enriquecer y fomentar notablemente la comunicación entre los médicos, sustituyendo de una forma más formal las charlas que se pueden desarrollar cuando se encuentran cara a cara.

3- La posibilidad de culminar la emisión en vivo (despedir al paciente) y continuar dialogando con sus colegas una vez emitida la consulta, permitiéndole así, concluir un diagnóstico compartido con el aval de todos o complementado por el criterio de otros médicos.

4- El médico que realiza la emisión -a diferencia del resto- tiene la posibilidad de modificar el cierre de la entrevista agregando información que considere relevante.

5- La posibilidad que tiene el médico emisor de publicar la entrevista en el momento que crea conveniente. Es decir, una vez culminada la emisión el médico tratante puede continuar con el caso días posteriores, agregando información al cierre de la

entrevista, considerando las opiniones de sus colegas, y en el momento que crea oportuno publicarlo, permitiendo así su consumo a demanda.

6- La posibilidad de guardar en sus equipos las consultas (entrevistas) a las que hayan sido invitados. Por ejemplo, si por alguna razón un médico no puede participar en una transmisión en vivo, puede luego que la consulta haya sido publicada, descargar la entrevista a su equipo. Por tratarse de una solución del estilo Smart Client, podrá ver el video asociado cuantas veces quiera, sin necesidad de estar conectado a Internet. Mejor aún, podrá acceder a un video con una calidad mejor a la que se observó durante la emisión en vivo. Además del video también tendrá toda la información asociada a una entrevista, incluida en la portada y cierre de la misma. Una situación similar puede darse con los médicos que la hayan presenciado: podrán verla nuevamente, observar detalles, adquirir conocimiento que fuera generado en forma colectiva.

7- La comodidad de poder realizar entrevistas en casi cualquier momento y lugar, independiente de tener que contar con una logística física y previa organización. No se necesita coordinar a las personas para que acudan a un mismo lugar en un mismo horario, con los problemas de transporte que esto significa. Sobre todo si se tiene en cuenta que los participantes pueden ser médicos que se encuentran en diferentes puntos del país. Es suficiente con agendar día y hora para la entrevista y que quienes deseen participar se conecten a la emisión en el momento estipulado.

Se forma de esta manera un ámbito de intercambio de conocimientos destacable desde todo punto de vista. Es decir, más allá de la posibilidad de presenciar en vivo una consulta de un determinado colega con un paciente, se refleja la importancia que tiene esto en este entorno de la medicina ya que para la evaluación y pronóstico del caso es necesario captar la reacción del paciente. Este es un caso particular que no se da en otras áreas de la medicina, en las cuales la entrevista con el paciente no tiene esta importancia.

En caso de que la entrevista no culmine con un diagnóstico, la misma sirve como experiencia en un caso clínico. Pueden realizarse comparaciones con otros casos, búsquedas de patrones o servir como base de conocimiento en el futuro.

El objetivo principal a cumplir fue el de modelar una *Cámara de Gesell* virtual, instaurando un ámbito en el que los miembros de esta comunidad pedan crear una base de conocimiento apoyada en el compartir información generada a partir de estas sesiones. Aplicando los conceptos y patrones Web 2.0 se presenta una solución que modela la realidad planteada permitiendo su aplicación no sólo en este entorno, sino en una amplia diversidad de disciplinas, denotando la potencia e importancia académica que este proyecto tiene.

Esta aplicación pensada inicialmente para el uso de un conjunto de médicos, ha sido implementada de manera abierta y generalizable. Fácilmente puede ser configurada para funcionar en otro ámbito de uso. Con un poco de visión pueden encontrarse otros ámbitos en donde la posibilidad de transmitir en vivo, contar con la funcionalidad de compartir mensajes durante una emisión, guardar información asociada a la emisión en un servidor y compartirla luego entre un grupo de usuarios, sean funcionalidades muy útiles. *G-Sell* puede ser usado en cualquier ambiente con estas necesidades.

6.2 Trabajo futuro

Durante el desarrollo del presente proyecto hubo consenso con la contraparte de *EviMed* al tomar ciertas determinaciones respecto al alcance que permitieran cumplir los objetivos planteados con la presentación de un prototipo terminado en un tiempo considerado razonable.

Aunque estas restricciones de tiempo determinaron un ajuste en el alcance, siempre se tuvo presente durante el proceso de diseño de la solución el hecho de poder cumplir con todos los requerimientos inicialmente planteados. Es por este motivo que se puntualizan a continuación algunos trabajos a realizar en futuros desarrollos con el fin de mejorar el prototipo presentado.

WikiMed: implantar en el ambiente de producción de *EviMed* la solución encontrada para la integración de la *WikiMed* con el sistema *Moodle*. (MediawikiSSO, AuthMoodle)

En lo referente a la aplicación *G-Sell* que promueve el uso de la *Cámara de Gesell,* se investigarán los siguientes puntos:

- *Picture in Picture*.
  Implementar esta funcionalidad a través de filtros DirectShow que manejen los flujos multimedia a nivel de frame. La decisión de programar directamente con el componente de terceros DirectShow fue considerando la posibilidad de ofrecer *Picture in Picture*.
- Dispositivos móviles.
  Se piensa investigar en esta rama ya que parece viable extender el producto a dispositivos móviles. A medida que el hardware baja de precio, los usuarios tienen más acceso a éstos y requieren más funcionalidades asociadas a los mismos.
- Seguridad de la información.
  Buscar la forma de incrementarla. A pesar de que la presente solución la considera a través de la encriptación de archivos y de la autenticación de usuarios tanto a nivel de aplicación, como de servicios Web, se marca que aún existen puntos vulnerables que deberían ser atendidos. La sensibilidad de los datos manejados requiere trabajar más en aspectos como la autenticación personalizada en el servidor de streaming (WMS) o el encriptado de la información guardada y obtenida desde la base de datos.
- Gestor de publicaciones y descargas.
  Implementar un gestor que permita la múltiple realización en forma simultánea de publicaciones y descargas de archivos, así como la posibilidad de reiniciar una descarga que no hubiera sido completada.
- Formato WMV9.
  Investigar y probar las posibilidades que brinda esta última versión del formato para archivos multimedia. Se han encontrado trabajos avanzados en la carga de perfiles para filtros definidos con este formato.
- Streaming multimedia.
  Investigar la posibilidad de desarrollar filtros DirectShow que permitan el uso MPEG4 transportado vía UDP.
- Chat.
  Estudiar la posibilidad de implementar el servicio de chat por medio de la programación con sockets.
- Mensajes de notificación.
  Permitir el envío de correo electrónico notificando a los usuarios invitados a presenciar una entrevista en vivo. Poder integrar esta funcionalidad con calendarios, para permitir agendar del lado del usuario la fecha y hora de realización de la entrevista.
- Otros dispositivos.
  Investigar en la posibilidad de conectar otro tipo de cámaras, así como también otros dispositivos de captura, como puede ser el caso de aparatos médicos. También el poder utilizar un video pre grabado durante la emisión en vivo.
- Incorporación de etiquetas.
  Consiste en que un usuario pueda asignar una o varias etiquetas a una entrevista. El programa deberá resolver un sistema de búsquedas por etiquetas, que complemente las búsquedas implementadas inicialmente. De esta manera se fomenta el uso de la información ya existente para casos nuevos. Además pueden ser usadas como parámetro de agrupamiento u ordenamiento de las entrevistas en las interfaces visuales, dando de esta forma, más personalización del sistema por usuario.

- Incorporación de búsquedas inteligentes.
  Además de las búsquedas implementadas y de la mencionada en base a etiquetas, es posible mejorar las búsquedas tratando de proveer una búsqueda inteligente. Llegando incluso a búsquedas ponderadas por diferentes criterios: etiquetas, fechas, usuario responsable de la emisión, información contenida en la emisión o tipos de archivo asociados, para el caso de usar varios dispositivos como punto de entrada.

6.3 Aprendizaje

Durante el transcurso del presente trabajo se exploró la educación a distancia. Esta rama, muy de moda actualmente, está fundamentada en las necesidades de las personas de armar comunidades de discusión, bases de conocimiento y sitios donde compartir información, simplemente valiéndose de la red y algunos requerimientos básicos como lo son un equipo, algunos componentes de software y una conexión a Internet.

Desde el punto de vista formativo, las investigaciones realizadas permitieron adquirir nuevos conocimientos acerca de las últimas tendencias en el uso de Internet, las preferencias de los usuarios y sus demandas a la hora de utilizar sistemas informáticos. Comprendimos mucho mejor el funcionamiento de las nuevas tecnologías y tendencias Web que permiten acompañar la formación y constante expansión de comunidades sociales con afinidad de intereses. Una de las mayores sorpresas que tuvimos fue constatar que Wikipedia pudiera generar conocimiento de forma tan válida como la propia Enciclopedia Británica. Esto nos dio una pauta muy clara de cómo funciona la inteligencia colectiva y la potencialidad que tiene.

Se estudiaron las características más relevantes de herramientas que no llegaron a usarse durante el desarrollo del proyecto. También se incursionó en la evaluación y uso de tecnologías y herramientas de liberación reciente, aplicando diferentes propuestas de las mismas.

Trabajamos en la instalación, creación, configuración y puesta en funcionamiento de máquinas virtuales, herramienta frecuentemente utilizada en simulaciones o incluso ambientes de producción, para contener distintos tipos de servidores aprovechando al máximo la potencia del hardware disponible. Fue precisamente sobre una máquina virtual que preparamos un ambiente LAMP (Linux Apache MySQL PHP) que pudiera servir como soporte para la Wiki instalada. Configuramos y probamos cada uno de estos componentes básicos con el cometido de dejar la Wiki operativa.

A lo largo del proyecto se trabajó con dos metodologías diferentes para alcanzar las metas planteadas, que nos permitió poner un sistema en producción en un caso y desarrollar un producto desde su inicio en otro.

La primera de ellas fue la aplicación de la filosofía de software libre, utilizada para implantar la Wiki. Es así que llegamos a vivir la experiencia de una puesta en producción en un ambiente real, con todas las precauciones que implica y enseñanzas que deja. Esta metodología de trabajo habitualmente brinda una solución muy buena ofreciendo una rápida puesta en marcha con poco esfuerzo, pero muchas veces no resulta sencillo adaptar estos componentes a necesidades particulares. Es así que bajo estas circunstancias debimos enfrentarnos a problemas difíciles de resolver por la escasa documentación disponible.

Con el fin de realizar los ajustes de configuración necesarios para personalizar la Wiki y migrar los datos de usuario desde la base existente incursionamos, aunque no en forma profunda, en el lenguaje PHP. En estos casos fue necesario comprender la lógica de cada programa para modificar el código PHP escrito por terceros. Realizamos scripts de modificación y carga de datos que resultaron de utilidad en repetidas oportunidades. También tuvimos la oportunidad de poner en práctica algunos conceptos y conocimientos de networking trabajando en forma supervisada, pero siempre en el ambiente real proporcionado por *EviMed*.

Por otro lado, ante la necesidad de presentar un producto que reuniese todos los requerimientos planteados en la Cámara de Gesell virtual y en vista de su estado del arte debimos realizar un desarrollo a medida.

En este caso pusimos en práctica una metodología de desarrollo cumpliendo etapas que permitieron diseñar tanto el sistema, como la aplicación, para posteriormente implementarla y realizar las pruebas correspondientes.

Para completar el producto final incorporamos bibliotecas desarrolladas por terceros, debiendo entender primero su funcionamiento para poder modificarlas y lograr las funciones que necesitábamos. Para esto fue necesario participar de foros y ponernos en contacto con desarrolladores o especialistas de otras partes del mundo (Noruega, Suecia, Grecia y Estados Unidos entre otros), para realizar consultas generales e intercambiar opiniones.

Incursionamos en el tratamiento de audio y video para plataformas Windows a través de las componentes DirectShow y DirectSound que forman parte de la librería DirectX. Aprender a trabajar con estas librerías incorporando los conceptos de filtro y grafo de filtros nos permitió construir nuestros propios modelos para la captura, grabación y emisión de contenidos multimedia. El hecho de manipular la información de flujos multimedia a nivel de cuadro abre un amplio abanico de posibilidades para la creación o transformación de imágenes, video y sonido en plataformas Windows.

También, fue necesario investigar sobre los distintos formatos de archivo multimedia y sus principales características.

Exploramos diversas alternativas que brinda Windows Media Services para la emisión de contenidos. Reforzamos y pusimos en práctica nuestros conocimientos teóricos de streaming a través de una red. Experimentamos con distintos anchos de banda y nos enfrentamos a las dificultades de transmitir video en vivo por Internet.

Una de las características de la solución construida fue la incorporación del concepto de Smart Client que permitiera a la aplicación de escritorio poder acceder a información aunque estuviese desconectada de Internet. Para cumplir con este objetivo trabajamos con bases de datos locales utilizando Sql Server Compact Edition.

Pensando en la creación y manipulación de la base de datos central del sistema, nos instruimos y aprendimos a trabajar con Linq to Sql que permitió modelar la base de datos relacional con clases de .NET. De esta forma fuimos capaces de consultar la base con Linq, así como actualizar, añadir y borrar datos de ella.

También rescatamos como experiencia enriquecedora el hecho de haber aprendido a escribir código para la plataforma .NET, más precisamente C sharp, trabajando en el entorno de desarrollo Microsoft Visual Studio que nos permitió crear la aplicación final y todos los servicios Web necesarios. Trabajamos aprovechando todas las ventajas que Windows Communication Foundation ofrece para la construcción de aplicaciones orientadas a servicios y creamos interfaces de usuario ricas y dinámicas a través de Windows Presentation Foundation.

Finalmente, queremos destacar el uso de Word como procesador de textos para la realización del presente informe que ha sido fruto de varios meses de trabajo en equipo.

Referencias

- [1] McLuhan, Marshall y Powers, Bruce; The Global Village: Transformations in World Life and Media in the 21st Century; Oxford University Press; 1989.
- [2] Castells, Manuel; La Sociedad Red (The Rise of Network Society), La Era de la Información; Volumen 1; Alianza; 1996.
- [3] Empresa EviMed – último acceso: 28/06/09 http://www.evimed.net/
- [4] Revista de Psiquiatría del Uruguay; Ginés, Angel; Educación médica continua y permanente; volumen 65, nº 2; octubre 2001 – último acceso: 30/06/09 http://www.mednet.org.uy/~spu/revista/oct2001/04_emc-4.pdf
- [5] Red social eviDoctor – último acceso: 28/06/09 http://www.evidoctor.net/
- [6] WikiMed – último acceso: 28/06/09 http://www.evimed.net/wiki/portada.htm
- [7] Definición de software libre http://www.gnu.org/philosophy/free-sw.es.html
- [8] Ghezzi, Carlo; Fundamentals of software engineering; New Jersey; Prentice-Hall; 1991.
- [9] MediaWiki - último acceso: 28/06/09 http://www.mediawiki.org/wiki/MediaWiki
- [10] Educación a distancia - último acceso: 30/06/09 http://www.sld.cu/libros/distancia/indice.html
- [11] Origen del concepto wiki – último acceso: 28/06/09 http://observatorio.cnice.mec.es/modules.php?op=modload&name=News&file=article&sid=378
- [12] WikiWikiWeb – Primera Wiki – último acceso: 28/06/09 http://c2.com/cgi/wiki?WikiWikiWeb
- [13] Wikipedia – último acceso: 28/06/09 http://www.wikipedia.org/
- [14] Revista Nature – último acceso: 30/06/09 http://www.nature.com
- [15] Enciclopedia Británica – último acceso: 28/06/09 http://www.britannica.com/
- [16] Giles, Jim; otras publicaciones http://www.ncbi.nlm.nih.gov/sites/entrez?db=pubmed&cmd=DetailsSearch&term=%22Giles+J%22&log$=activity
- [17] Página oficial de DokuWiki – último acceso: 30/06/09 http://www.dokuwiki.org/dokuwiki
- [18] WikiMedia - último acceso: 28/06/09 http://www.wikimedia.org/
- [19] Página oficial de OpenWikiNG – último acceso: 30/06/09 http://www.openwiki.com/ow.asp?OpenWikiNG
- [20] Página oficial de TikiWiki – último acceso: 30/06/09 http://info.tikiwiki.org/tiki-index.php
- [21] Comparativa WikiMatrix - último acceso: 30/06/09 http://www.wikimatrix.org
- [22] Moodle – último acceso: 28/06/09 http://moodle.org/
- [23] MediaWikiSSO – último acceso: 30/06/09 http://moodle.org/file.php/5/moddata/forum/366/300692/mediawiki_sso.tgz

- [24] Gerner, Jason; Naramore, Elizabeth; Owens, Morgan; Warden, Matt; Professional LAMP: Linux, Apache, MySQL and PHP Web Development; Wrox; 2005.
- [25] Biografía Arnold Lucius Gesell – último acceso: 28/06/09 http://www.answers.com/topic/arnold-gesell
- [26] Manual de uso de las Cámaras de Gesell – último acceso: 28/06/09 www.poder-judicial.go.cr/secretariadegenero/documentacion_relevante/CAMARAS%20DE%20GESSELL.doc
- [27] Messenger – último acceso: 28/06/09 http://download.live.com/messenger
- [28] Google Talk – último acceso: 28/06/09 http://www.google.com/talk/intl/es/
- [29] Oovoo – último acceso: 28/06/09 http://www.oovoo.com/
- [30] Acrobat.com – último acceso: 28/06/09 http://www.Acrobat.com
- [31] VSee – último acceso: 28/06/09 http://vsee.com/
- [32] SightSpeed – último acceso: 28/06/09 http://www.sightspeed.com/
- [33] PalBee – último acceso: 30/06/09 http://www.palbee.com/
- [34] MeBeam – último acceso: 30/06/09 http://www.mebeam.com/
- [35] Vidivic – último acceso: 30/06/09 http://www.vidivic.com
- [36] MegaMeeting – último acceso: 30/06/09 http://www.megameeting.com/
- [37] Skype – último acceso: 28/06/09 http://www.skype.com
- [38] PCWorld – Big Skype Update Coming Next Month – último acceso: 27/06/09 http://www.pcworld.com/article/156744/big_skype_update_coming_next_month.html
- [39] SnapYap – último acceso: 30/06/09 http://www.snapyap.com
- [40] Video Conferencia: Las Mejores Herramientas De Videoconferencia Que Todos Pueden Usar - Sharewood Guide – último acceso: 30/06/09 http://www.masternewmedia.org/es/2008/09/12/video_conferencia_las_mejores_herramientas_de_videoconferencia.htm
- [41] Vuze – último acceso: 28/06/09 http://www.vuze.com
- [42] Elluminate – último acceso: 28/06/09 http://www.elluminate.com/
- [43] Polycom – último acceso: 28/06/09 http://www.polycom.com/
- [44] Smart Client – último acceso: 28/06/09 http://msdn.microsoft.com/en-us/library/ms998439.aspx#offline-ch01_smartclientsandofflinecapabilities
- [45] Flex – último acceso: 28/06/09 http://www.adobe.com/products/flex/
- [46] Flash – último acceso: 28/06/09 http://www.adobe.com/products/flash/
- [47] Adobe AIR – último acceso: 28/06/09 http://www.adobe.com/es/products/air/
- [48] d2B Network – Design To Business – último acceso: 28/06/09 http://www.d2bnetwork.com/

- [49] Smart Client Architecture and Design Guide – último acceso: 28/06/09 http://msdn.microsoft.com/en-us/library/ms998506.aspx

Glosario

.Net	Plataforma de Microsoft orientada a la creación de software para Internet.
AAC	Es un formato de audio digital comprimido más estable que MP3 y que además tiene mejor calidad, produciendo un audio más cristalino.
API	Interfaz de programación de aplicaciones. Es el conjunto de funciones y procedimientos que ofrece cierta biblioteca para ser utilizado por otro software como una capa de abstracción.
Apple Inc.	Es una empresa estadounidense de tecnología informática. Fundada en 1976, tuvo un rol prominente en la revolución informática de los años 70, aunque en la actualidad la cuota de mercado mundial de computadoras Macintosh es del 8%, Apple se jacta de tener siempre los diseños más elegantes e innovadores del mercado, como lo demuestran sus últimos productos como el iMac, iPod, MacBook o el iPhone.
ASF	Formato avanzado de transmisión por secuencias.
Buffer	Espacio en memoria donde se almacenan datos. Evita que el programa o el recurso que lo requiera, se quede en algún momento sin datos.
Buffering	Necesidad de seleccionar datos para editarlos o procesarlos antes de llevarlos a un archivo o base de datos.
Booleanos	Datos que pueden tomar los valores verdadero o falso.
CMS	(Content Management System) Sistema de gestión de contenidos. Programa que permite que los participantes generen un framework para la creación y administración de contenidos, principalmente en páginas web.
Codecs	Describe una especificación desarrollada en software, hardware o una combinación de ambos, capaz de transformar un archivo con un flujo de datos (stream) o una señal. Pueden codificar el flujo o la señal y recuperarlo o descifrarlo del mismo modo para la reproducción o la manipulación en un formato más apropiado para estas operaciones.
Codificador	Término empleado para hacer referencia al usuario que inicia la transmisión en vivo (encoder).
Download	Nombre de carpeta utilizada en la aplicación *G-Sell* para la descarga de archivos.
DTD	(Document Type Definition), es una descripción de estructura y sintaxis de un documento XML. Su función básica es la descripción del formato de datos para usar un formato común y mantener la consistencia entre todos los documentos que utilicen el mismo tipo de definición.
E-book	Versión electrónica de un libro o publicación que ha sido confeccionada para ser comercializada en Internet.
E-learning	Medio electrónico para el aprendizaje a distancia.
Entrevista	Término usado para referirse a la consulta realizada entre el paciente y el médico emisor.
Emisor	Médico que realiza la consulta en vivo con el paciente.
eviDoctor	Plataforma educativa para la red social de la empresa EviMed.

EviMed	Nombre de la empresa para la cual se realizó el proyecto.
Frame	O marco, es una 'porción' de la página principal en la que se carga otra página distinta. Sirve para tener la página separada en partes.
Framework	Estructura de soporte definida, mediante la cual otro proyecto de software puede ser organizado y desarrollado. Puede incluir soporte de programas, bibliotecas, lenguaje interpretado.
G-Sell	Denominación usada para referenciar la aplicación creada 'Cámara de Gesell'.
Gesell	Nombre de la base de datos creada para la aplicación.
Grilla	Es un instrumento para ordenar elementos gráficos del texto y de las imágenes.
Ide	Un entorno de desarrollo integrado o, en inglés, Integrated Development Environment. Es un programa compuesto por un conjunto de herramientas para un programador.
Infoware	Término que define sitios web como por ejemplo Amazon.com que permiten compartir datos (comentarios, puntuaciones) y como resultado devuelve la creación de valor (opiniones de otras personas sobre un libro que desea comprar).
Interoperabilidad	Característica de los computadores que permite su interconexión y funcionamiento conjunto de manera compatible, con independencia del sistema operativo utilizado o la arquitectura del sistema.
iPod	Es un reproductor de música digital con pantalla a color y disco duro o memoria flash, creado por Apple. Son conocidos por su interfaz de fácil manejo basada en el uso de una sola "rueda táctil", como la única forma de navegación.
iTunes store	Es un negocio de Apple de venta en línea de archivos de música y video que son accesibles a través del software iTunes que ellos diseñaron y distribuyen gratuitamente. Las tiendas de Apple permiten una forma muy sencilla de adquirir canciones y transferirlas a un dispositivo iPod.
LAN	Se trata de una red local compuesta por al interconexión de varios equipos y periféricos. Su extensión está limitada físicamente, por ejemplo, a un edificio.
Mail	Correo electrónico, medio de comunicación que permite el envío y recepción de mensajes escritos.
Máquina virtual	Software que emula a un equipo y puede ejecutar programas como si fuese un equipo real.
Meme	Unidad teórica mínima de transmisión de herencia cultural, difusión cultural que se propaga de una mente a otra de la misma manera en que un gen se propaga de un organismo a otro.
Metadata	Son datos que describen otros datos Por ejemplo en una biblioteca se especifican autores para buscar libros, así los metadatos ayudan a ubicar datos.
MP3	MPEG-1 audio layer 3, es un formato de audio digital que se convirtió en el estándar utilizado para flujos y compresión de audio de alta calidad.
MPEG-4	Es un grupo de estándares de codificación y compresión de audio y video normalizado por el grupo MPEG (Moving Picture Experts Group). Su uso principal son los flujos de medios audiovisuales.
Multicast	Multidifusión. Es el envío de la información en una red a múltiples destinos simultáneamente.

Multimedia	Es un término que se aplica a cualquier objeto que usa simultáneamente diferentes formas de contenido informativo como texto, sonido, imágenes, animación o video para informar o entretener al usuario.
On-line	en línea, significa que se debe estar conectado a Internet.
Pestaña	También nombrada lengüeta. Es un elemento de la interfaz de un programa que permite cambiar rápidamente lo que se está viendo sin cambiar de ventana.
Picture in Picture	Pequeña ventana superpuesta a la ventana principal al mismo tiempo.
Podcast	Portable On Demand Broadcast, es decir, emisión portátil a solicitudes. Es un archivo de audio que se baja a la computadora mediante un enlace directo o desde la suscripción a un canal RSS.
Podcasting	Consiste en la creación de archivos de sonido -generalmente en formato MP3 o AAC-o de video (podcasts) y su distribución mediante un archivo RSS que permite suscribirse y descargarlo de Internet para que el usuario lo escuche cuando quiera.
Postear	Poner a disposición del público cierta información en Internet.
Ranking	Clasificación de mayor a menor útil para establecer criterios de valoración.
Receptor	Se refiere al médico que consume la entrevista en vivo a través de la aplicación. Es decir aquel que no se encuentra físicamente frente al paciente.
Rollover	Protocolo que soporta variedad de versiones de Players. Es utilizado por el reproductor de Windows Media Player.
RSS	Es un sencillo formato de datos que es utilizado para redifundir contenidos a suscriptores de un sitio web. El formato permite distribuir contenido sin necesidad de un navegador, utilizando un software diseñado para leer estos contenidos. RSS es parte de la familia de los formatos XML desarrollado específicamente para todo tipo de sitios que se actualicen con frecuencia.
Script	Es un archivo de procesamiento por lotes o programa de simple comprensión que generalmente se almacena en un archivo de texto plano y se utiliza para realizar tareas administrativas como combinar componentes e interactuar ya sea con el sistema operativo, como con los usuarios.
Sindicación	(de contenidos), la distribución y publicación de un mismo contenido en muchos sitios.
SSL	(Secure Socket Layer), son protocolos criptográficos que permiten comunicaciones seguras en Internet. Proporcionan autenticación y privacidad de la información entre extremos sobre Internet mediante el uso de criptografía.
Software	Programas de computadoras. Instrucciones responsables de que el hardware (máquina) realice su tarea.
Streaming	Término que se refiere a ver u oír un archivo directamente en una página web sin necesidad de descargarlo antes al equipo.
String	Es una cadena de caracteres de longitud variable.
Suite ofimática	Conjunto integrado de programas que son comúnmente utilizados en oficinas y que cada uno cumple funciones específicas en las tareas laborales.

TCP	Protocolo de control de transmisión, es uno de los principales protocolos de la capa de transporte del modelo TCP/IP.
Tmeout	Tiempo máximo que debe esperar un programa para conectarse antes de abortar su tarea.
Testing	Pruebas de software. Procesos que permiten verificar y revelar la calidad de un producto de software.
Ubicuidad	Capacidad de estar presente en todas partes simultáneamente, estar en la Web y poder ser encontrado y visto.
UDP	Protocolo de datagrama de usuario, es un protocolo no orientado a conexión de la capa de transporte del modelo TCP/IP.
Unicast	Es el envío de información desde un único emisor a un único receptor.
Usabilidad	Técnicas que ayudan a los seres humanos a realizar tareas en entornos gráficos de computadoras.
URL	(Uniform Resource Locator), es la cadena de caracteres con la cual se asigna una dirección única a cada uno de los recursos disponibles en Internet. Existe un único URL para cada página de cada documento de la web.
Videoconferencia	Comunicación simultánea bidireccional de audio y video que permite mantener reuniones con grupos de personas situadas en lugares diferentes.
VoIP	Voz sobre Protocolo de Internet, también llamado Voz sobre IP, VozIP, VoIP. Es un grupo de recursos que hacen posible que la señal de voz viaje a través de Internet empleando un protocolo IP.
Web Services	Son un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones. Distintas aplicaciones de software desarrolladas en diferentes lenguajes de programación y ejecutados sobre cualquier plataforma pueden utilizar servicios web para intercambiar datos en redes de computadores.
Web semántica	Es una web extendida dotada de mayor significado en la que cualquier usuario en Internet podrá encontrar respuestas a sus preguntas de forma más rápida y sencilla gracias a una información mejor definida. Se basa en la idea de añadir metadatos semánticos a la www.
Web 2.0	Es la transición que se ha dado a aplicaciones tradicionales hacia aplicaciones que funcionan a través de la web enfocadas al usuario final. Fomenta la colaboración y el intercambio ágil de información entre los usuarios de una web.
Web 3.0	Se trata de la evolución del uso y la interacción en la red a través de diferentes caminos. Se asocia con el término Web semántica, añade significado a la Web. Incluye la transformación de la red en una base de datos.
WikiMed	Nombre de la wiki creada para *EviMed*.
www	La World Wide Web es un medio de comunicación de texto, gráficos y otros objetos multimedia a través de Internet.
XML	(eXtensible Markup Language), es un lenguaje utilizado para hacer referencia a otros lenguajes de etiquetas y permite definir la gramática de lenguajes específicos. Se propone además como un estándar para el intercambio de información estructurada entre diferentes plataformas.

XSD	(XML Schema Definition), es un lenguaje de esquema utilizado para describir la estructura y las restricciones de los contenidos de los documentos XML.

Índice de ilustraciones

SECCIÓN I

1. Introducción

Figura 1.1: Enfoque orientado a software libre usado en WikiMed.....4
Figura 1.2: Modelo cascada realimentado utilizado en el desarrollo de G-Sell.....4
Figura 1.3: Organización del documento.....6
Figura 1.4: Cronograma comparativo.....7

SECCIÓN II

Parte A – WIKIMED

2. Desarrollo de la investigación

Figura A2.1: Comparativa entre principales alternativas para Wiki.....10

4. Implementación

Figura A4.1: Página de inicio.....15
Figura A4.3: Editar redacción de un artículo.....16
Figura A4.4: Grabar página.....16
Figura A4.5: Editar discusión de un artículo.....16
Figura A4.6: Artículo editado.....17

5. Pruebas y resultados

Figura A5.1: Ayuda en WikiMed.....19

Parte B - CÁMARA DE GESELL

2. Desarrollo de la investigación

Figura B2.1: Ejemplo Cámara de Gesell.....21

4. Implementación

Figura B4.1: Arquitectura de acceso a datos.....34
Figura B4.2: Arquitectura de subsistema de chat.....35
Figura B4.3: Arquitectura de subsistema de emisión.....36
Figura B4.4: Arquitectura de publicación y descarga de archivos.....37
Figura B4.5: Arquitectura general de la solución.....37
Figura B4.6: Tablas de la base de datos Gesell.....39
Figura B4.7: Componente de acceso a datos.....41
Figura B4.8: Formato archivo XML.....44
Figura B4.9: Acceso a archivos.....46
Figura B4.10: Definición de Metadata.....48
Figura B4.11: Diagrama de clases del componente local.....53
Figura B4.12: Presentación de G-Sell.....55
Figura B4.13: Grafo de captura y emisión.....56



Universidad de la República



Facultad de Ingeniería
Instituto de Computación
Montevideo – Uruguay

Proyecto de Grado

Aplicación de redes sociales a la educación a través de Internet.

ANEXOS AL INFORME FINAL

Alejandra Borghi
Yanet Cortés
Walter López Da Cunha

Tutores:
Martín Furno
Pablo Rebufello

Informe de Proyecto de Grado presentado al Tribunal Evaluador como requisito de graduación de la carrera Ingeniería en Computación.

Índice de anexos

A1 – Anexos referidos al estado del Arte

1. Web Semántica....2
1. Introducción....2
2. La web hoy....2
3. Concepto....2
4. Objetivo....3
5. Funcionalidades....3
6. Arquitectura de la Web Semántica....3
6.1 Nivel de recursos....4
6.2 Nivel sintáctico....4
6.3 Nivel de Descripción de Recursos....4
6.4 Nivel de ontologías....4
6.5 Resto de niveles....5
7. Máquinas vs. significado....5
7.1 Requerimientos....5
8. Mapa conceptual de la Web Semántica....7
9. Integración: Web Service....7
10. Componentes....8
11. Mejoras introducidas....8
12. En un futuro....9
13. Ejemplos....9
14. Caso práctico....10
15. Referencias....10
2. Tendencia Web 2.0....11
1. Web 2.0....11
1.1 Utilidad....11
1.2 Significado....11
2. Principios....11
2.1. La www como plataforma de trabajo....11
2.2. Fortalecimiento de la inteligencia colectiva....12
2.3. Gestión de base de datos como competencia básica....12
2.4. El fin del ciclo de las actualizaciones del software....12
2.5. Programación ligera y búsqueda de la simplicidad....12
2.6. El software no limitado a un solo dispositivo....12
2.7. Experiencia enriquecedora del usuario....13
3. Comparación con los Servicios de la Web 1.0....13
3.1 Relación entre empresas....13
3.2 Diferencias entre sus sitios web....14
3.3 Diferencias claras entre ambas Webs....15
3.4 Algunos ejemplos visuales....15
4. La Web 2.0 en la educación....17
5. Mapa MEME....17
6. Exponentes....17
6.1 Blogs....18
6.2 Wiki....18
6.3 Edublogs....18
6.4 Folcsonomía....18
7. Tecnologías que apoyan a la Web 2.0....18
7.1 Tecnologías que dan vida a un proyecto....18
8 Ventajas....19
8.1 Extrapolación y sindicación absoluta....19
8.2 Aplicaciones como servicio y no como producto....19
8.3 Ubicuidad....19
8.4 Cooperación....19

8.5 Seguridad....19
9. Desventajas....20
9.1 Confidencialidad....20
9.2 La conexión a Internet....20
10. Conclusiones....20
11. Referencias....21
3. Funcionalidades Web 2.0 de Genexus....22
1. Introducción....22
2. Genexus Rocha....22
3. User Controls....23
3.1 Generalidad....23
3.2 Controles ya definidos....23
3.3 Creación de nuevos controles....25
4. Próxima versión....28
5. Referencias....29
4. Estudio de iTunes U....30
1. Introducción....30
2. Requerimientos de sistema....30
3. Organización y acceso de la información....30
4. iTunes U como tecnología educacional....31
5. Caso de uso....31
5.1 Visión institucional....32
5.2 Visión del usuario....34
6. Referencias....37
7. Apéndice La aplicación iTunes....38

A2 – Anexos referidos a tecnologías y patrones de diseño

1. Ocho patrones básicos para el desarrollo de la Web 2.0....40
1. Aprovechar la inteligencia Colectiva....40
2. Los datos son el nuevo "Intel inside"....40
3. Innovar en el ensamble....41
4. Experiencias ricas de usuario....41
5. Software sobre el nivel de un único hardware....42
6. Beta perpetuo....42
7. Aprovechar la larga cola....43
8. Modelos livianos y escalabilidad rentable....43
9. Referencias....43
2. Modelo-Vista-Controlador....44
1. Concepto....44
2. Objetivo....44
3. Descripción....44
4. Elementos del Patrón....44
4.1 Modelo....44
4.2 Vista....45
4.3 Controlador....45
5. Flujo de control....45
6. Referencias....45
3. Introducción a Windows Presentation Foundation (WPF)....46
1. Descripción....46
2. Ventajas....46
2.1 Plataforma unificada para interfaces de usuario modernas....46
2.2 Posibilidad de trabajar juntos desarrolladores y diseñadores....47
2.3 Tecnología común para interfaces de usuario de Windows y explorador web.50
3. Diseño y controles....50
3.1 MediaElement....51
3.2 StackPanel....51
3.3 DockPanel....52
3.4 Canvas....53

3.5 Grid....55
3.6 FlowDocumentPageViewer....56
3.7 Tab Control....57
3.8 Wrap panel....58
3.9 Binding de controles con datos....59
3.10 Crear estilos para controles....59
4. Referencias....61
4. Introducción a Windows Communication Foundation (WCF)....62
1. Principios de diseño....62
2. Concepto....62
3. Abc de WCF....62
3.1 Address....62
3.2 Bindings....62
3.3 Contract....62
4. Servicios....63
5. Behaviours....63
6. Referencias....64
5. Publicación y descarga de archivos....65
1. Introducción....65
2. Cliente....65
2.1 Transferencia de entrevistas....67
2.2 Transferencia de archivos comunes....70
3. Servidor....70
3.1 Configuración....70
3.2 Contrato....70
3.3 Implementación....71
3.4 Destino de archivos....73
4. Librería DotNetZip....74
4.1 El namespace Ionic.Zip....74
5. Referencias....75
6. ChatMessage....77
1. Objetivo....77
2. Generalidades....77
3. Implementación....77
7. Servicios dúplex....79
1. Concepto....79
2. Generalidades....79
3. Contrato dúplex....79
4. Creación del servicio para Cámara de Gesell....79
4.1 Declaración de las interfaces....79
4.2 Implementación del servicio....80
4.3 Generación de proxies....81
4.4 Invocación del servicio desde la aplicación cliente....82
5. Referencias....83
8. Chat del canal del mismo nivel....84
1. Concepto....84
2. Aplicación....84
3. Generalidades....84
4. Terminologías....84
5. Implementación....84
6. Referencias....85

A3 – Anexos referidos a Base de Datos

1. Base de Datos....87
1. Entidades....87
2. Tablas....88
2.1 Users....88
2.2 Specialities....88

2.3 Publications....88
2.4 Interviews....89
2.5 MetadataInfo....89
2.6 ChatUsers....90
2.7 ChatMessage....90
2.8 Entities....90
2.9 Configuration....90
2.10 Metadata....91
2.11 UsersSpecialities....91
2.12 InterviewsGuests....91
3. Referencias....92
2. Acceso a datos....93
1. Arquitectura....93
2. Linq to Sql....93
2.1 Modelo....93
2.2 Carga....94
2.3 Transaccionalidad....94
2.4 Serialización....95
2.5 Seguridad....95
3. Modelo de datos....95
3.1 Diseñador relacional de objetos....95
3.2 Generación automática....97
4. Extensión del modelo de datos....98
4.1 Clases parciales....98
4.2 Métodos....99
5. Subcapas de acceso a datos....100
6. Operaciones sobre los datos....101
6.1 Obtener datos....102
6.2 Alta de datos....103
6.3 Modificar datos....104
6.4 Eliminar datos....106
7 Sintaxis de Linq to Sql....106
7.1 Palabras reservadas....106
7.2 Ejemplos....107
8. Referencias....107
3. Componente Local....109
1. Archivos locales....109
2. Sql Server Compact Edition....109
2.1 Sintaxis....110
3. Base de datos local....111
3.1 Estructura....111
4. Acceso a datos....113
4.1 Componente....114
4.2 Transferencia....115
5. Referencias....116

A4 – Anexos referidos a video y streaming

1. Captura y streaming multimedia....118
1. Introducción....118
2. Construcción del grafo....118
2.1 Construcción del grafo de captura....118
2.2 Construcción del grafo de emisión....121
3. Presentación de captura en WPF....126
3.1 Modificación a librería MediaKit....126
3.2 Audio con DirectSound....131
3.3 Captura imagen para portada....132
4. Presentación de emisión en vivo en WPF....133
5. Presentación de entrevista grabada en WPF....133

6. Referencias....133
2. Multimedia en Windows....134
1. Librerías DirectX....134
1.1 Historia....134
1.2 Componentes....135
1.3 Funcionamiento interno....135
2. Librerías DirectShow....136
2.1 Evolución de DirectShow....136
2.2 Filtros y streams multimedia....137
2.3 Modelo de aplicación....138
2.4 Tipos de multimedia (MediaType)....139
2.5 El Protocolo de conexión....141
2.6 Protocolo de streaming....142
2.7 Control del flujo....143
3. Librería DirectShowNET....144
3.1 Interfaces....144
4. Librería MediaKit....147
4.1 Objetos COM y DispatcherObject....147
4.2 Interfaces y clases base....147
4.3 Controles....148
4.4 Últimos avances....149
5. Referencias....149
3. Formato de archivos multimedia....150
1. Introducción....150
2. Windows Media Format SDK....150
3. Formato Sistema Avanzado (ASF)....151
4. Librería WindowsMediaNET....152
4.1 Interfaces....152
4.2 Perfiles....154
5. Formato MPEG-4....155
6. Referencias....155
4. Servidor de streaming....156
1. Introducción....156
1.1 Descripción de streaming de video....156
1.2 Tecnología detrás del streaming de video....156
1.3 Limitaciones a considerar....158
2. Protocolos de transmisión....158
2.1 Transmission Control Protocol (TCP)....158
2.2 User Datagram Protocol (UDP)....158
3. Protocolos de streaming....159
3.1 Protocolo de transporte en tiempo real (RTP)....160
3.2 Protocolo de Microsoft Media Server (MMS)....160
3.3 Protocolo Shoutcast/Icecast (ICY)....161
3.4 Streaming a través de firewalls....161
4. Windows Media Services9 Series....162
4.1 Introducción....162
4.2 Streaming....163
4.3 Configuración de servicio Windows Media....163
4.4 Características....164
4.5 Instalación....166
4.6 Rendimiento....169
4.7 Planificación de la capacidad de red....171
5. Referencias....172

A5 – Anexos referidos a los resultados

1. Experiencia en WikiMed....174
2. Bosquejo de interfaces en papel....187
1. Solapa Emitir....187

2. Solapa Emisiones on-line.....191
3. Solapa emisiones grabadas.....192
3. Internet encyclopaedias go head to head.....194

A1 – Anexos referidos al estado del Arte

1. Web Semántica

1. Introducción

Se describirá en el presente apéndice los aspectos más relevantes que diferencia la World Wide Web (www) de una web dotada de significado como lo es la Web Semántica. Qué es la Web Semántica, qué es lo que motiva su aparición, qué mejoras aporta, cómo funciona, para qué sirve, qué problemas resuelve al tratarse de una web extendida y basada en el significado.

2. La web hoy

Existen hoy mil maneras diferentes de referirse a lo mismo, es decir distintas entidades con el mismo significante, lo que conlleva a una ambigüedad y a una falta de precisión.

Los datos no se encuentran inmediatamente disponibles para ser procesados por otro software, no se encuentran interconectados.

Los buscadores entienden las palabras pero no las relaciones, se arrojan miles resultados con poca precisión, resultados sensibles al vocabulario, etc.

En todos los casos se puede extraer datos, pero no información ni menos aun conocimiento. La solución a estos inconvenientes es: Web Semántica.

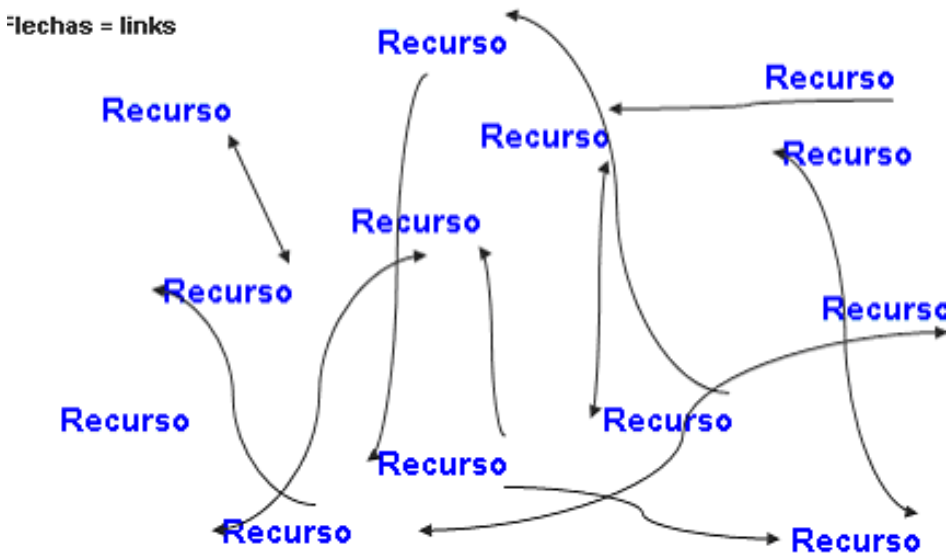


Figura 1: Cómo funciona la web hoy.

3. Concepto

La Web Semántica es un Web extendida se basa en la idea de añadir metadatos (datos que hablan de datos) semánticos a la World Wide Web.

Esas afirmaciones adicionales deben ser proporcionadas de manera formal para que así puedan ser evaluadas automáticamente por máquinas de procesamiento.

Dotada de mayor significado permite que cualquier usuario en Internet pueda encontrar respuestas a sus preguntas de forma más rápida y sencilla gracias a una información mejor definida.

Esta Web extendida y basada en el significado, se apoya en lenguajes universales que resuelven los problemas ocasionados por una web carente de semántica en la que, en ocasiones, el acceso a la información se convierte en una tarea difícil y frustrante.

4. Objetivo

Su objetivo es crear un medio universal para el intercambio de información basado en representaciones del "significado de los recursos de la Web", de una manera "descifrable" para las máquinas.

Así se mejoraría Internet ampliando de esta manera la interoperabilidad (condición mediante la cual sistemas heterogéneos pueden intercambiar procesos o datos.) entre los sistemas informáticos y se reduciría la participación de operadores humanos en los procesos de flujo de información inteligentes.

Resumiendo, los objetivos deseados serían:

- Buscar establecer un sentido a los contenidos
  Facilitar a las máquinas la comprensión de lo que queremos representar.
- Permitir una conexión entre los recursos de la web
  Extender los hiperenlaces a relaciones semánticas.
- Gestión "inteligente" de los recursos
  Búsquedas de información. Extracción de contenidos. Integración de recursos heterogéneos.
- Interoperabilidad Semántica.
- Tener una web más útil
  El botón "voy a tener suerte" de Google.

En suma, el objetivo de la Web Semántica es que la Web pase de ser una colección de documentos a convertirse en una base de conocimiento.

5. Funcionalidades

La Web Semántica ayuda a resolver dos grandes problemas existentes hoy: sobrecarga de información con el consiguiente problema de interoperabilidad.

Permite a los usuarios delegar tareas en software, gracias a la semántica en la Web, el software es capaz de procesar su contenido, razonar con este, combinarlo y realizar deducciones lógicas para resolver problemas cotidianos automáticamente.

6. Arquitectura de la Web Semántica

La nueva red semántica ha sido estructurada por niveles, estableciendo una jerarquía de abstracción y unas dependencias entre los distintos niveles.

Actualmente esta arquitectura no ha sido totalmente descrita, se encuentra al día de hoy en el nivel de la lógica.

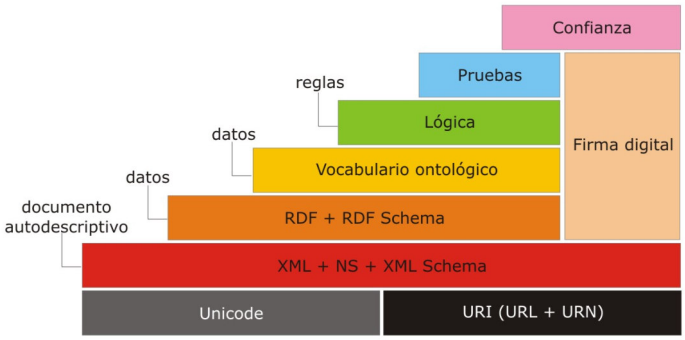


Figura 2: Arquitectura de la Web semántica.

6.1 Nivel de recursos

Es el primer nivel, aquí se incluyen la identificación de recursos web, estableciendo así la principal importancia que tiene definir el conjunto de recursos distribuidos por la red.

Los URIs (Identificador de Recurso Uniforme) identifican de forma inequívoca un recurso introducido en la red, este identificador además cumplirá la función de identificador de objetos en el mundo real.

6.2 Nivel sintáctico

En este nivel se soluciona el problema de cómo definir distintos lenguajes de etiquetados para añadir contenido semántico a las páginas web. En él se encuentra XML (eXtensible Model Language) un metalenguaje, un lenguaje para escribir lenguajes que permite formalizar y validar los documentos, estructuración y capacidad de ampliación.

6.3 Nivel de Descripción de Recursos

En este nivel se encuentra RDF (Resource Description Framework) -Marco de Descripción de Recurso- es una recomendación de la World Wide Web Consortium (W3C), un consorcio internacional que produce estándares para la World Wide Web. Este marco está diseñado para estandarizar la definición y el uso de las descripciones de metadatos de los recursos web.

También se utiliza para representar datos simples. El área donde está enmarcado es la descripción de los recursos de la red, por ejemplo: personas, dispositivos, páginas.

RDF es un lenguaje etiquetado basado en XML, además ofrece estructuras formales (no ambiguas) gracias al uso de URIs, permitiendo la codificación, intercambio y el procesamiento automatizado de metadatos normalizados.

6.4 Nivel de ontologías

OWL (Ontology Web Language) es oficialmente según W3C la forma más apropiada de compartir conocimiento en la web.

Técnicamente OWL se descompone en tres sublenguajes incrementales para ser utilizado según las necesidades:

OWL Life

Es el más sencillo de los tres. Cubre las necesidades de clasificar conceptos jerárquicamente y con restricciones sencillas, permitiendo sólo cardinalidad 0 o 1.

OWL DL

DL fue diseñado para aquellos usuarios que requiriesen la máxima expresividad del lenguaje pero teniendo en cuenta la eficiencia computacional.

OWL full

Es el más completo de los tres sublenguajes, permitiendo la misma expresividad que OWL DL y con toda la capacidad sintáctica que ofrece RDF, sin embargo añade una gran carga de procesamiento.

6.5 Resto de niveles

- El nivel de la lógica pretende dar flexibilidad a la arquitectura para realizar consultas e inferir conocimiento a partir de las ontologías de la capa anterior.
- El nivel de seguridad permite asignar grados de confianza y seguridad a los distintos recursos distribuidos en la web a través de firmas digitales, redes de confianza u otras técnicas de autenticación por red. Al día de hoy, no existe una recomendación oficial por parte de W3C acerca del resto de capas de la arquitectura propuesta.

7. Máquinas vs. significado

La manera que se tiene para que las máquinas entiendan significados es:

- Hacer máquinas más inteligentes --> Enfoque IA (Agentes Inteligentes).
- Hacer datos más inteligentes --> Enfoque Ws (Web Semántica).

Esto se logra con los siguientes requerimientos:

7.1 Requerimientos

La Web semántica requiere:

- Metadatos
  Etimológicamente, datos que hablan de datos por lo que ayudan a clasificarlos. Datos acerca de estructura y contenido de los documentos de la web, sean html, imágenes, video, audio, documento de texto u otros.
- Ontologías
  Las ontologías se refieren a un dominio de información, un área de conocimiento determinada. Vocabularios que las máquinas pueden entender, especificados de manera precisa para permitir diferenciar términos y poder referenciarlos de manera precisa y unívoca.
- Datos inteligentes, esto significa :
  Independientes de las aplicaciones, clasificados, parte de un ecosistema mayor de información (ontología).

Y para aprovechar todo su potencial

- Lógica y motores de inferencia.
- Agentes inteligentes (IA).

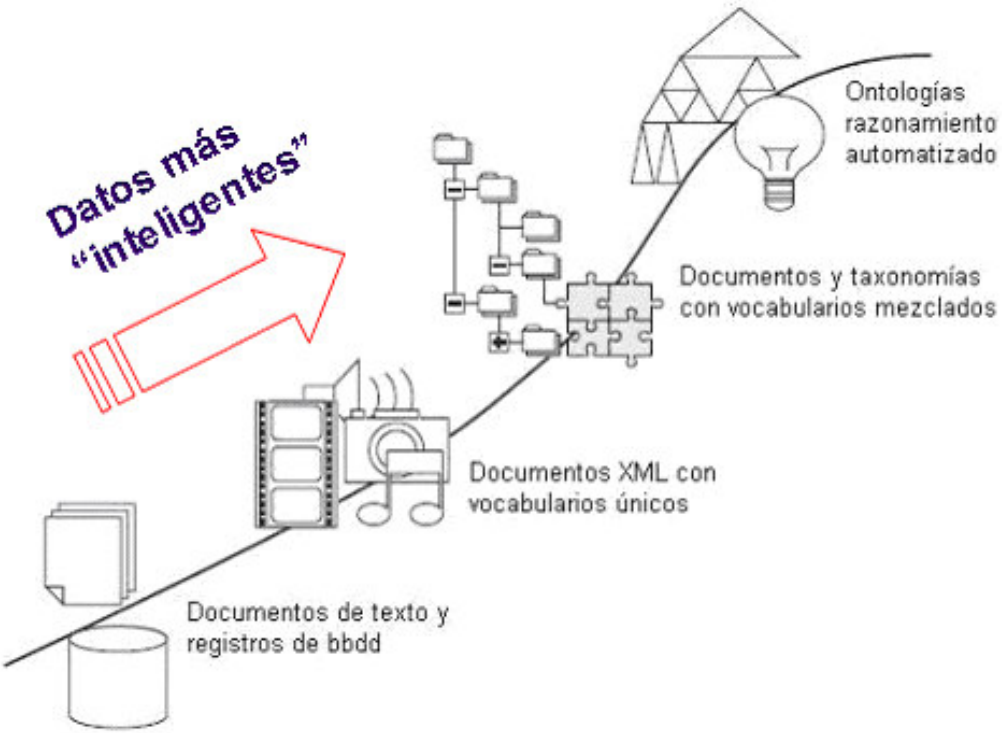


Figura 3: Datos en la Web semántica.

El papel de los Agentes Inteligentes

Ellos son los usuarios de una Web Semántica ya que un agente inteligente entiende (lo que se pide), comprende (el contenido de los sitios), valida (si lo encontrado corresponde a lo pedido) y deduce (nueva información sobre la ya obtenida).

Pero, para entender una consulta, comprender sitios web, validar información y deducir nueva información se precisa estandariza:

- El alfabeto.
- Las referencias.
- El lenguaje.
- El formato.
- Las anotaciones sobre significados.
- Los conceptos generales.
- Las reglas y sistemas de deducción.

Y éstas son, precisamente, las funciones que aportan las distintas capas (niveles) de la Web Semántica.

En la Web Semántica el agente podrá:

- Leer los contenidos semánticos de una página en un lenguaje formal (OWL).
- Utilizar conocimiento compartido (ontologías).
- Descubrir exactamente donde están los recursos que requieren (URIs).

Los agentes y aplicaciones web que exploten este conocimiento anotado en páginas web serán capaces de interpretar el conocimiento ontológico y axiomas para realizar tareas como búsquedas inteligentes y el mantenimiento de ontologías y de los recursos, etc.

8. Mapa conceptual de la Web Semántica

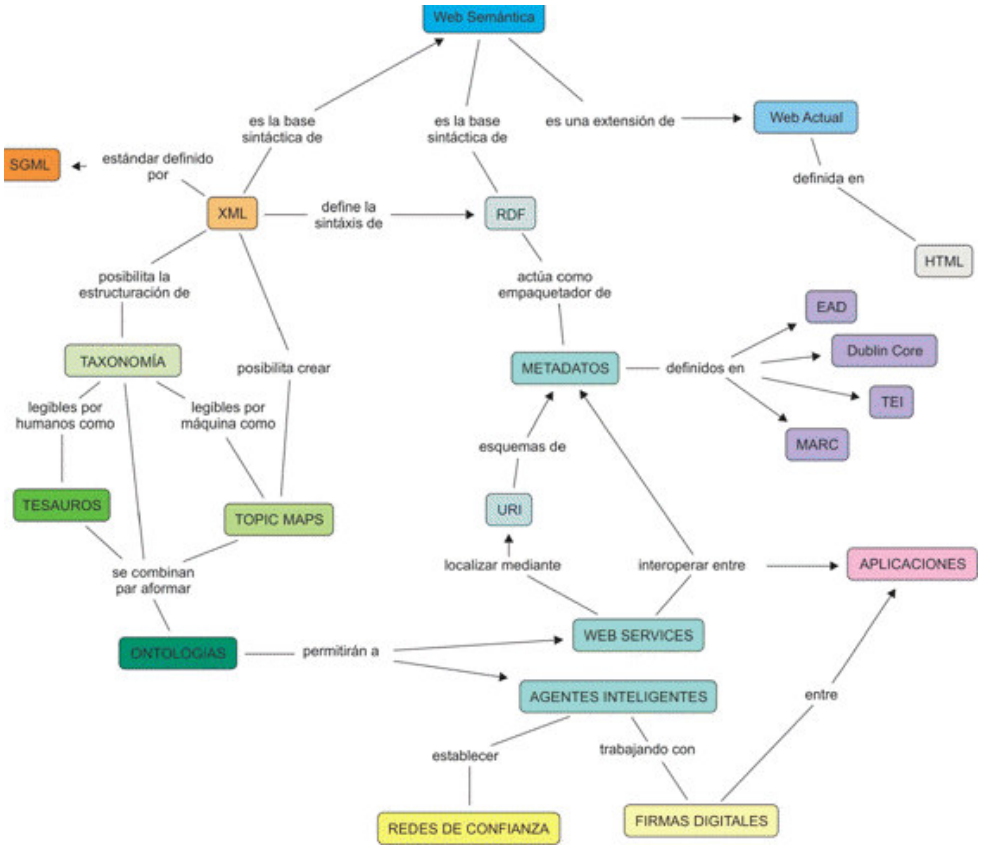


Figura 4: Mapa conceptual de la Web Semántica.

9. Integración: Web Service

Es un conjunto de aplicaciones o de tecnologías con capacidad para interoperar en la Web, éstas intercambian datos entre sí con el objetivo de ofrecer servicios.

Los proveedores ofrecen sus servicios como procedimientos remotos y los usuarios solicitan un servicio llamando a estos procedimientos a través de la Web.

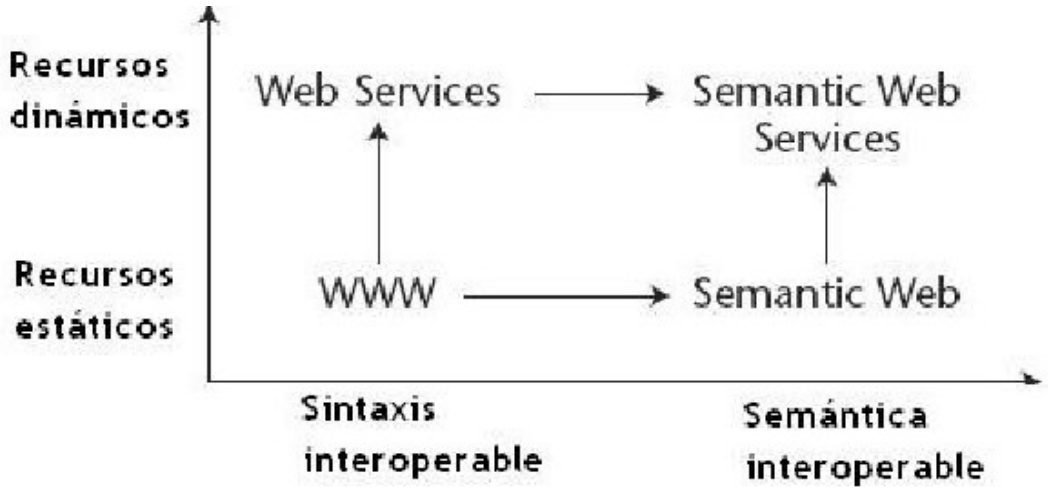


Figura 5: Integración Web Services con Web Semántica.

10. Componentes

Los principales componentes de la Web Semántica son los metalenguajes y los estándares de representación XML, XML Schema, RDF, RDF Schema, OWL. Se los detallan a continuación:

- XML: otorgará la sintaxis para documentos estructurados, pero no agrega semántica.
- XML Schema: restringe la estructura de documentos XML y extiende a XML con datatypes.
- RDF: es un modelo de datos para objetos (recursos) y relaciones entre ellos. Provee semántica simple para este modelo de datos, y puede ser representado con sintaxis de XML.
- RDF Schema: es un vocabulario para describir clases y propiedades de recursos RDF, usando semántica para jerarquías generalizadas de esas propiedades y clases.
- OWL: agrega vocabulario para describir propiedades y clases: entre otros, relaciones entre clases, cardinalidad, igualdad y características de propiedades.

11. Mejoras introducidas

La Web Semántica introdujo una serie de mejoras desde varios puntos de vista, algunas de ellas son:

- Máquinas que comprendan significados.
- Discriminación automática de relevancias.
- Interoperabilidad.
- Búsqueda más eficiente: resultados certeros en menos tiempo.
- Éxito para encontrar lo que se busca.
- Una red de significados.
- Información clasificada.
- Jerarquía de datos.
- Infraestructura ordenada.
- Convertir a la red en una mega base de datos.



Figura 6: Cómo funciona la Web Semántica.

La Web semántica implica una infraestructura común, mediante la cual se pueda compartir, procesar y transferir información de forma sencilla.

Dicha infraestructura hace que las máquinas comprendan el significado de la información lo que facilita y agiliza el trabajo humano.

12. En un futuro

La Web semántica permitirá:

- Realizar deducciones lógicas.
- Realizar inferencias.
- Combinar información.
- Generar información 'nueva' a partir de la existente.
- Realizar consultas complejas.
- Automatizar procesos.

13. Ejemplos

Se describen a continuación algunos ejemplos de aplicación de Web Semántica desde el punto de vista de los usuarios y desde el punto de vista de las aplicaciones.

De usuarios

- "El viaje perfecto": avión, hotel, restaurante, visa, trámites para llevar mascotas, vacunas.
- "El mini-asesor contable": impuestos, fechas de pago, lugares, bancos, noticias.
- "Una agenda médica": cartilla de la obra social, sacar turno, procesar receta, pedir los lentes.

De aplicación

- RSS
  Es un vocabulario RDF basado en XML que permite la clasificación de información (noticias y eventos) de tal manera que sea posible encontrar información precisa adaptada a las preferencias de los usuarios.
  Los archivos RSS contienen metadatos sobre fuentes de información especificadas por los usuarios cuya función principal es avisar a los usuarios de que los recursos que ellos han seleccionado para formar parte de esa RSS han cambiado. Notifican de forma automática cualquier cambio que se realice en esos recursos de interés seleccionados.
- FOAF
  Es un proyecto de Web Semántica, que permite crear páginas Web para describir personas, vínculos entre ellos y cosas que hacen y crean.
  Se trata de un vocabulario RDF, que permite tener disponible información personal de forma sencilla y simplificada para que pueda ser procesada, compartida y reutilizada.
  Dentro de FOAF podemos destacar FOAF-a-Matic, una aplicación JavaScript que permite crear una descripción FOAF de uno mismo. Con esta descripción, los datos personales serán compartidos en la Web pasando a formar parte de un motor de búsqueda donde será posible de una forma rápida y sencilla descubrir información a cerca de una persona en concreto y de las comunidades de las que es miembro.

14. Caso práctico

Dos hermanos tienen que ajustar su calendario para acompañar a su madre a unas sesiones de rehabilitación ordenadas por los médicos. Para ello utilizan un agente electrónico que revisa sus agendas para las próximas semanas y las combina con las horas disponibles en los centros de rehabilitación más cercanos cubiertos por el seguro médico familiar.

En pocos minutos consiguen programar de nuevo sus agendas, incluyendo las visitas de su madre, y resuelven el problema. Y lo hacen gracias "no a la World Wide Web de hoy sino a la Web Semántica en la que se convertirá mañana".

Hoy en día, la mayor parte del contenido que hoy aparece en Internet está diseñado para ser leído por las personas, no para que lo puedan manipular con sentido unos programas informáticos. Para que esto sea una realidad será suficiente con que los contenidos se introduzcan en las páginas web utilizando un programa de escritura de Webs semánticas.

Así, lo que hasta ahora había sido básicamente "un medio de documentos para personas" pasará a ser un sistema de datos e información que se podrán procesar automáticamente. Por ejemplo, será posible que "los agentes de software que deambulan de una página a otra puedan realizar tareas sofisticadas para los usuarios".

15. Referencias

- Ejemplos de Web Semántica - último acceso: 29/06/09
  http://www.w3c.es/Divulgacion/Guiasbreves/WebSemantica#ejemplo
- Futuro de Internet: Web Semántica - último acceso: 29/06/09
  http://www.baquia.com/com/20010523/art00008.print.html
- Guía breve de la Web Semántica - último acceso: 29/06/09
  http://www.w3c.es/Divulgacion/Guiasbreves/WebSemantica
- Hacia la Web Semántica - último acceso: 29/06/09
  http://www.hipertexto.info/documentos/web_semantica.htm
- La Web semántica problemática y arquitectura - último acceso: 29/06/09
  http://www.cii-murcia.es/informas/abr05/articulos/
  La_web_semantica_problematica_y_arquiectura.php
- Web Semántica - último acceso: 29/06/09
  http://web-semantica.org/
- W3C - último acceso: 29/06/09
  http://www.w3c.es/Presentaciones/2005/1018-WebSemanticaREBIUN-MA/

2. Tendencia Web 2.0

1. Web 2.0

Se trata de un fenómeno tecnosocial que se hizo popular a través de sus aplicaciones más representativas como lo son Wikipedia, YouTube, Flickr, WordPress, Blogger, MySpace, Facebook, OhMyNews y de las muchas ofertas de cientos de herramientas, tratando de captar usuarios que sean generadores de sus propios contenidos.

En Web 1.0 principalmente se trata el estado estático, los datos que se encuentran en ella no se actualizan, no existe ningún tipo de interacción con los usuarios. Para cambiar estos aspectos es que surgen los servicios de la Web 2.0.

Se la podría definir también como la evolución de las aplicaciones tradicionales hacia aplicaciones web enfocadas al usuario final, no es precisamente una tecnología, sino que es la 'actitud' con la que debemos trabajar para desarrollar en Internet.

Principalmente lo que se buscó fue la sustitución a las aplicaciones de escritorio en muchos de sus usos. Se trata de aplicaciones que generan colaboración y dan servicios, lo que tendería a reemplazar las aplicaciones de escritorio.

1.1 Utilidad

Teniendo en cuenta la renovación y evolución de los proyectos es allí donde se destaca la verdadera utilidad del Web 2.0.

Siempre se está trabajando en renovar y mejorar algunos proyectos porque se cree firmemente que la única constante debe ser el cambio, y en Internet sobre todo, el cambio debe de estar presente con más frecuencia.

1.2 Significado

La forma más fácil de comprender su significado es a través de ejemplos. Se puede comparar servicios Web que marcan claramente la evolución hacia la Web 2.0 con una "nueva forma de hacer las cosas".

2. Principios

Se describen a continuación siete principios constitutivos de las aplicaciones Web 2.0.

2.1. La www como plataforma de trabajo

Las aplicaciones Web 2.0 comienzan a convivir con un modelo de negocio de paquetes: con derechos de propietarios, que deben utilizarse en forma legal, comprar los derechos de uso y actualizar el software cada cierto tiempo con los consiguientes costos adicionales por la licencia.

Las nuevas compañías ofrecen software gratuito, utilizando la web como plataforma, de esta forma las herramientas y sus contenidos existen en la propia web y no en la computadora del usuario.

Junto a la Wikipedia, el peer to peer de Napster fue una de las herramientas que más contribuyó a transformar el modelo de contenidos centralizados hacia uno en que los usuarios se convirtieran en servidores.

A diferencia de Napster, donde el disco duro del usuario guardaba la información y el software de la aplicación en línea permitía compartirla, las herramientas Web 2.0 utilizan su servidor para almacenar la información, y el usuario conectado a la red siempre tiene acceso a ella. La idea clave a todo esto es: "el producto mejora con el uso y con el crecimiento de una comunidad que no cesa de subir contenidos".

2.2. Fortalecimiento de la inteligencia colectiva

La dinámica de la inteligencia colectiva está creciendo de la mano de esta nueva generación de aplicaciones web, con software en línea diseñado a partir de una interfaz de fácil uso, escalable de **valor añadido** en contenidos y de acceso gratuito.

En el entorno de la Web 2.0 los usuarios actúan de la manera que deseen, en forma tradicional y pasiva, navegando a través de los contenidos, o en forma activa, creando y aportando sus propios contenidos.

2.3. Gestión de base de datos como competencia básica

Este principio tiene una palabra clave: 'infoware' software más datos.

Lo valioso de las aplicaciones Web 2.0 son los datos, ya que en muchos casos el software es un recurso abierto o de fácil implementación.

Lo que se busca es obtener una masa importante de usuarios que produzcan un volumen de datos de gran valor, poseer esta información clave y usuarios "co-desarrolladores" otorga valor comercial al producto.

2.4. El fin del ciclo de las actualizaciones del software

Se rompe el modelo inicial del software cerrado con derecho de uso y bajo el principio de la obsolescencia planificada, para pasar al uso del software como servicio gratuito, corriendo en la propia web y en combinación con los datos.

Los paquetes de software tradicionales siguen siendo productos costosos que en poco beneficia a los consumidores, las nuevas aplicaciones Web 2.0 se desarrollan con el objetivo de sustituir a estos productos y ganar clientes dispuestos a producir y subir los datos a nuevas plataformas.

2.5. Programación ligera y búsqueda de la simplicidad

Los modelos de programación ligera ponen énfasis en la reducción de la complejidad, donde "menos es más", se evita así las excesivas especificaciones y funciones del software empaquetado.

Se pretende que las aplicaciones crezcan sin complicaciones para el desarrollador y además que el usuario pueda ver los contenidos en la plataforma que desee a través de la sindicación y no cuando el desarrollador/proveedor disponga en su plataforma propietaria.

El concepto de actualización ha perdido valor en los productos de la Web 2.0, esto se ve reflejado en los blogs y wikis y en su propia lógica de creación de contenidos en forma dinámica.

2.6. El software no limitado a un solo dispositivo

La utilización de los productos de la Web 2.0 no se limita a las computadoras. Los teléfonos móviles de tercera generación (3G) comenzaron a ocupar nuevos espacios. Hoy además de teléfonos son plataformas para el entretenimiento, la gestión de la información y el fortalecimiento de las redes sociales.

Se ha asignado un rol importante en la industria a los dispositivos móviles en su convergencia con las aplicaciones Web 2.0.

Por otra parte la sindicación evita tener que navegar por los sitios originales, facilitando la tarea de lectura de los contenidos previamente seleccionados por el usuario. Por ejemplo, productos como Bloglines o Google Reader facilitan la tarea.

2.7. Experiencia enriquecedora del usuario

Antes la web era sólo contenido textual y gifs animados, luego apareció Flash Macromedia para darle al usuario una experiencia más generosa a nivel gráfico.

Con la aparición de las aplicaciones Web 2.0 el usuario se ve enriquecido en cuanto a experiencia e intercreatividad, se logran interfaces con capacidad de acceso en todo lugar y momento a los servicios Web, con usabilidad, familiaridad y sencillez de las interfaces de los sistemas operativos.

Si se piensa en contenidos dinámicos sucede lo mismo, herramientas clásicas para producir web como Dreamweaver y Flash quedan atrás de las aplicaciones diseñadas en Ajax, Ruby on Raits o BitTorrent para los productos Web 2.0.

Si se buscan sinónimos que engloben los siete principios constitutivos, hay muchos que quedan relegados por no ser inclusivos de todas sus propiedades.

Si se habla de 'redes sociales' vemos que se trata de un conjunto mayor a las aplicaciones Web y que existen cientos de ellas que no respetan todos los principios recién mencionados, como por ejemplo, cualquier Intranet.

Para cerrar esta polémica se cree que el valor referencial y reproductivo de los nuevos conceptos está dado por su sencilla comprensión para 'públicos no–expertos', por la practicidad en el uso y por su sentido amplio. Éstos son los principales valores del término Web 2.0, aunque exista cierta saturación en su uso.

3. Comparación con los Servicios de la Web 1.0

Las notorias diferencias existentes en su esquema de funcionamiento son presentadas a continuación:

3.1 Relación entre empresas

La siguiente tabla presenta la relación que existe entre empresas características de la Web 1.0 y de la 2.0.

"Web 1.0"	"Web 2.0"
DoubleClick: Era un pionero de los Web Services, sin embargo se vio limitado por su modelo de negocios. Sus ofertas requieren un contrato formal de ventas que limita su mercado.	**Google AdSense:** "Servicios Publicidad Inteligente". Se trata de publicaciones, anuncios digitales, que hace coincidir los anuncios con el contenido del sitio del cliente. Le permite a los usuarios tener un seguimiento del rendimiento de su publicidad, (ver ejemplo en la sección 3.4 Google AdSense).
Ofoto: "Servicio Fotográfico On-line". Permite subir imágenes en JPEG, compartir álbumes de fotos, posibilita tener videos en línea, etc.	**Flickr:** "Comunidades Fotográficas". Es un servicio que ayuda a publicar, organizar y compartir fotos en línea (ver ejemplo Flickr).
Akamai: Ha creado un entorno de funcionamiento digital para la web, consta de una plataforma global de miles de servidores.	**BitTorrent:** "Distribución de Contenidos". Es un protocolo que básicamente es Peer to Peer. Adopta el enfoque de la descentralización de Internet. Cada cliente es también un servidor.

Mp3.com: Es un sitio web manejado por Redes de CNET que proporcionan la información sobre la música digital, artistas, canciones, servicios, comunidad y tecnologías.	**Napster.com:** Es el mejor directorio de MP3 en Internet. Sirve para conseguir gratis e ilimitadamente cualquier cantante o tema a nivel mundial.
Britannica On-line: Enciclopedia Británica cuenta con 75.000 artículos completos. Contiene libros de referencia y geografía mundial, permite navegar por categorías.	**Wikipedia:** "Enciclopedias". Es una enciclopedia de contenido libre que todos pueden editar.

3.2 Diferencias entre sus sitios web

"Web 1.0"	"Web 2.0"
Sitios Web Personales: Se crean y publican páginas propias, sirven para compartir intereses, gustos, imágenes, etc. Cuando hablamos de sitio web se dice que es todo aquél espacio con contenido de protocolo HTTP.	**Blogs:** "Bitácora". Algo así como un libro de navegación, un blog entra en la categoría de página web, se trata de sitios web que manejan entradas organizadas cronológicamente y por categorías. Una o más personas publican contenidos de forma periódica los que pueden ser comentados, discutidos, aprobados y desaprobados por visitantes del blog.
Especulación con dominios Web: La forma más fácil de identificar un sitio web es a través de su dirección de dominio. El .com nos familiariza con los nombres de dominio de Internet y nos permite conectar con los servidores web que alojan sitios web.	**Optimización de herramientas de búsqueda SEO:** "Search Engine Optimization". Es una solución que permitirá a una página web lograr la posición más alta en la lista de resultados de buscadores, la optimización se basa en una serie de criterios de búsqueda, etc.
CMS: "Content Management System". Sistemas de Gestión de Contenidos, es un programa que permite generar una estructura de soporte para la creación y administración de contenidos por parte de los participantes en páginas web.	**Wikis:** "Administradores de Contenidos". Una Wiki sirve para crear páginas web de forma rápida y eficaz.
Directorios, clasificación y taxonomías: Un directorio web es un tipo de sitio web que contiene un directorio organizado de enlaces con otros sitios web con una estructura de categorías y subcategorías.	**Tagging à "folksonomías":** Los tags son palabras o frases que se utilizan para categorizar fuentes de noticias una vez que se haya suscripto a ellas, (ver ejemplo TAGs).

Screen Scraping: "Raspado de pantalla". Es una técnica de programación que consiste en tomar una presentación de una información para, mediante ingeniería inversa, extraer los datos que dieron lugar a esa presentación.	**Servicios Web:** Es un conjunto de protocolos y estándares que sirven para intercambiar datos entre aplicaciones.
Stickness: Tiempo que los usuarios pasan conectados.	**Sindicación:** Es un avance muy significativo, se trata de la distribución y publicaciones de un mismo contenido en muchos sitios.

3.3 Diferencias claras entre ambas Webs

"Web 1.0"	"Web 2.0"
Web como fuente de información.	Web de participación.
Páginas HTML estáticas.	Páginas dinámicas o interactivas.
Páginas raramente actualizadas.	Páginas actualizadas por los usuarios.
Enfoque hacia lo visual.	Enfoque hacia los contenidos.
Páginas sencillas. Cualquier usuario podría hacer una web.	Los usuarios utilizan aplicaciones ya diseñadas, por lo que la creación de algo similar sería complicada para no expertos.

3.4 Algunos ejemplos visuales

Google AdSense

"Anuncios de Textos".

Anuncio en la parte superior izquierda



Anuncio en la parte superior derecha



Figura 1: Ejemplos de anuncios visuales.

Flickr

Mapa mostrando fotos geoposicionadas.

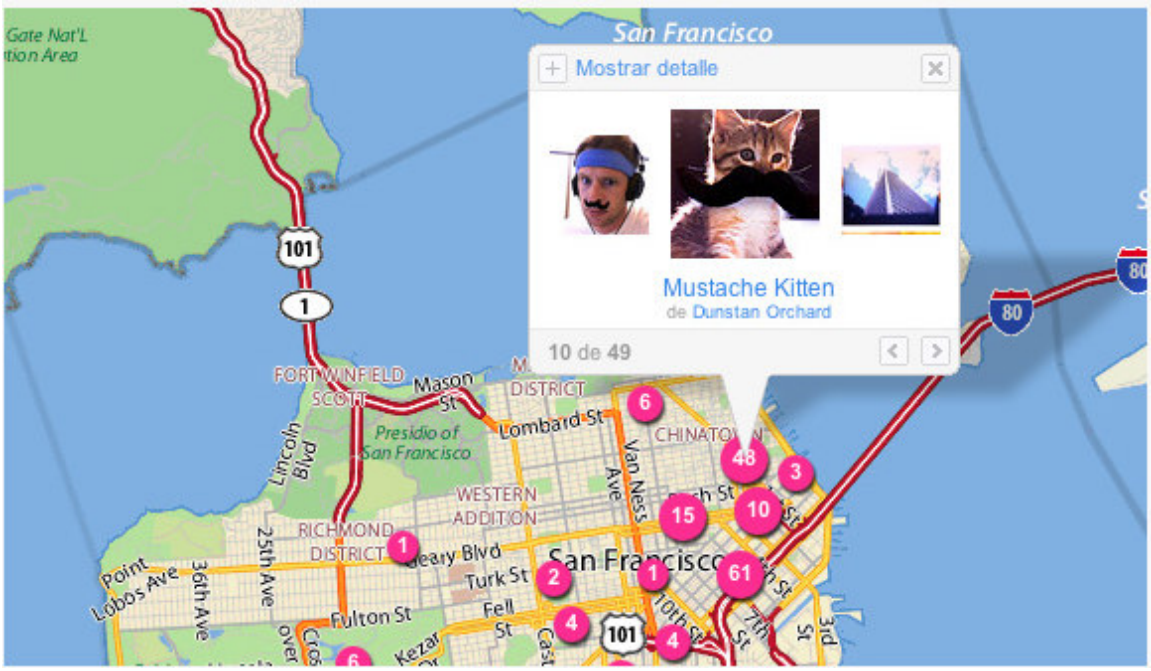


Figura 2: Fotos geoposicionadas.

TAGs

Se despliega la información de todas las "fuentes marcadas" en dicho tag.

Figura 3: Fuentes marcadas en tag.



4. La Web 2.0 en la educación



Figura 4: Aplicación de la Web 2.0 en la educación.

5. Mapa MEME

Por el presente mapa podemos visualizar las muchas ideas que irradian desde el núcleo de la Web 2.0.

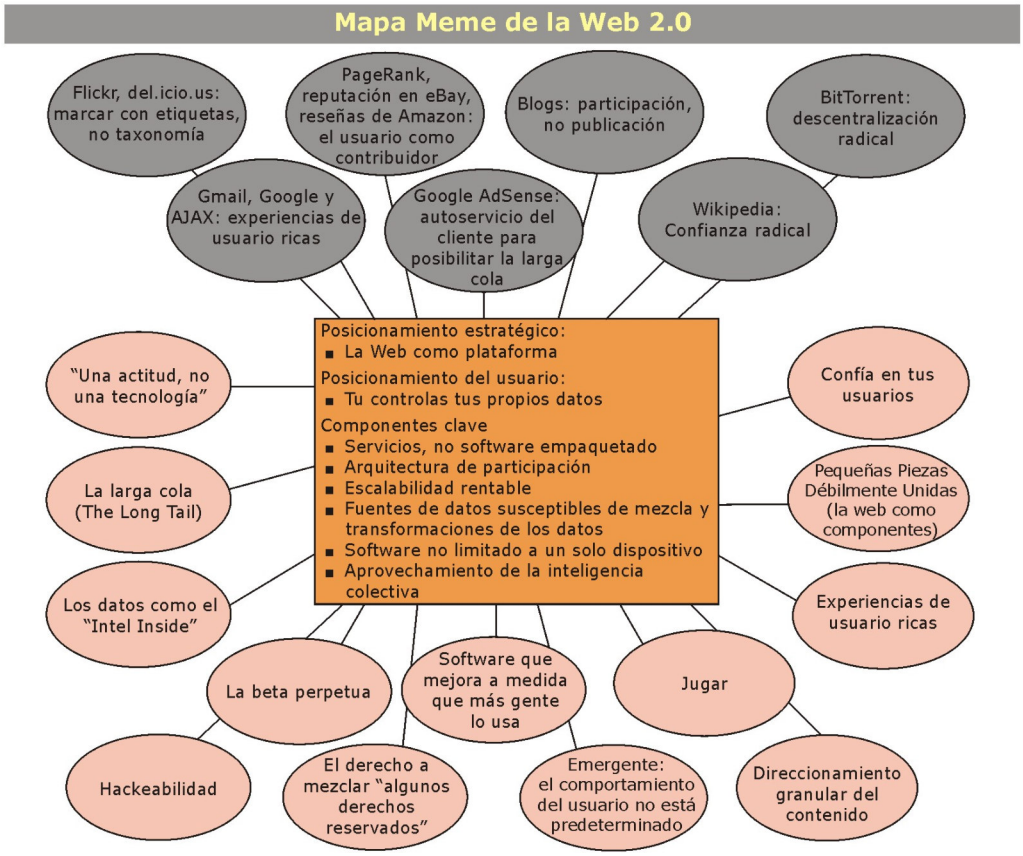


Figura 5: Mapa MEME.

6. Exponentes

Se describen a continuación algunos de los exponentes más destacados de esta nueva Web 2.0. Ellos son Blogs, Wiki, Edublogs y Folcsonomías.

6.1 Blogs

En el mundo de los blogs se encuentra el **Trackback** (palabra difícil de traducir al español) es una de las características más subutilizadas aquí.

Permite la comunicación entre blogs, se trata de un mecanismo de notificación entre websites, también llamado retroenlace. Sirve para enviar una referencia de un blog 'A' a un blog 'B' sobre un tema que tengan en común. Su utilidad se ve reflejada en el 'debate' que se genera, formándose así una red de conocimiento.

El **PingBack** es un sistema diferente del TrackBack en cuanto a su funcionamiento, aunque pretende exactamente lo mismo. Está basado en el autodescubrimiento del enlace.

Si un blog 'A' referencia a otro 'B' y este otro 'B' tiene activado el sistema de PingBack es él quien se encargará de hacer notar que el blog 'A' ha colocado un enlace hacia el 'B'. Por lo tanto no hace falta enviar un ping hacia el blog 'B', solo basta esperar que éste descubra el enlace que le referencia desde el blog 'A'.

6.2 Wiki

Una Wiki sirve para crear páginas web de forma rápida y eficaz, su finalidad es permitir que varios usuarios puedan crear páginas web sobre un mismo tema, de esta forma cada usuario aporta un poco de su conocimiento para que la página web sea más completa, creándose así una comunidad de usuarios que comparten contenidos acerca de un mismo tema o categoría.

6.3 Edublogs

Es una forma fácil de poder escribir periódica, personal o colectivamente en Internet.

Se permite el debate o los comentarios sobre cada uno de los temas o mensajes que se vayan tratando.

6.4 Folcsonomía

Es un tipo de sistema de clasificación distribuida. Generalmente es creada por un grupo de individuos. Actualmente existen muchas herramientas que se apoyan en los conceptos de Redes Sociales y son ellas las que incluyen las folcsonomías como criterio organizativo.

Por ejemplo, **Flickr** es una herramienta Web gratuita que permite almacenar on-line archivos fotográficos, pero que va más allá: se puede asignar títulos a los mismos, incluir comentarios, permitir que los demás también lo hagan generando así un debate entorno a una imagen, asignarles tags, crear círculos de personas que comparten los mismos tags, etc.

7. Tecnologías que apoyan a la Web 2.0

Existen varias tecnologías que se están utilizando actualmente y que deberían de examinarse con cuidado para seguir evolucionando junto a la web.

7.1 Tecnologías que dan vida a un proyecto

- Transformar software de escritorio hacia la plataforma de la web.
- Respeto a los 'estándares' como el 'XHTML'.
- Separación de contenido del diseño con uso de "hojas de estilo".

- 'Sindicación' de contenidos.
- 'Ajax' (java script asincrónico y xml).
- Uso de 'Flash', Flex o Lazlo.
- Uso de 'Ruby on Rails' para programar páginas dinámicas.
- Utilización de 'redes sociales' al manejar usuarios y comunidades.
- Dar control total a los usuarios en el manejo de su información.
- Proveer APIs o XML para que las aplicaciones puedan ser manipuladas por otros.
- Facilitar el 'posicionamiento' con URL sencillos.

8 Ventajas

Existen varias ventajas que ofrece la Web 2.0. Algunas de ellas son detalladas a continuación

8.1 Extrapolación y sindicación absoluta

El hecho de que todas las aplicaciones se realicen sobre Web, va a permitir que entre ellas se pueda compartir toda la información.

- Propagación inmediata de contenido e información (RSS) que va a permitir un mejor desarrollo de la estructura en red.
- Uso de otras fuentes para desarrollar nuevas, esto permitirá que el desarrollo de nuevas aplicaciones se centren en la aportación de **valor añadido**, centrando los recursos en lo nuevo, y sacando partido de lo hecho por otros.

8.2 Aplicaciones como servicio y no como producto

- Esto elimina el coste de acceso de las PYMES (pequeñas y medianas empresas) a la tecnología más moderna, anulando las barreras de entrada a competir en los mercados por esa vía.
- El pago se hace por servicio, lo que en pequeñas empresas es una muy pequeña cantidad en relación al coste de la plataforma que obtiene y fuera de su alcance a través de inversiones que las sacarían del mercado.
- Es un servicio externalizado, sin barreras de salida para el cambio.
- Las innovaciones son constantes sin necesidad de actualizaciones ni cambios.

8.3 Ubicuidad

La Web ya se ha consagrado como el canal de interoperabilidad por excelencia. Las aplicaciones basadas en Web pueden desarrollarse en cualquier terminal (y no necesariamente en los PC): ordenadores, móviles, TV digital, etc. Esto va a permitir tener la información en todo momento y desde cualquier terminal con conexión a Internet.

8.4 Cooperación

Las necesidades de la sociedad radican principalmente en la cooperación entre los diferentes actores. Estos sistema permiten anular prácticamente los costes para ello (incluido los costes de relaciones sociales), llegando incluso a la cooperación entre actores que no mantienen relación. La estructura del sistema coordinará la relación.

8.5 Seguridad

A nivel de PYMES la capacidad de seguridad y de protección de datos de servidores de empresas profesionales será siempre mucho mayor que la mantenida en servidores compartidos o en los mismos PCs de gestión diaria.

Pérdidas de datos por fallos del sistema, virus, ataques, etc., son constantes en los equipos personales sin que se mantengan copias de seguridad adecuadas y siendo el coste de restauración muy elevado para estas empresas.

9. Desventajas

Así como presenta ventajas, la nueva Web 2.0 tiene también sus desventajas. Se detallan algunas a continuación.

9.1 Confidencialidad

Al estar almacenados en servidores ajenos, la seguridad de datos confidenciales, como la contabilidad y la facturación, son algunos de los aspectos que más se debaten.

Teniendo en cuenta las necesidades de las PYMES, es probable que los datos estén en mejor 'cuidado' de servidores de empresas dedicadas a ello que en PCs que normalmente son mucho más vulnerables a ataques de virus, troyanos y espías.

Es un problema que se debe solventar con buenas prácticas y concientización por parte de las empresas.

9.2 La conexión a Internet

La dependencia del sistema a la conexión de Internet sigue siendo una barrera a su adopción.

Si bien las empresas de telecomunicaciones cada vez son más fiables y mantienen mejores conexiones, siempre existe la posibilidad de quedarse sin conexión en la mitad de una jornada laboral, lo que impediría el uso del sistema. No obstante, este problema tiende a convertirse en tan residual como la caída de la energía eléctrica, que si bien ocurre, no es causa suficiente como para no operar en base a ella.

10. Conclusiones

Podemos notar entonces que la Web 2.0 no es un cambio tecnológico, si no que se trata de un cambio en la filosofía con que los usuarios y empresas usan Internet.

Los usuarios quieren:

- Participar (blogs, wikis).
- Poder elegir (AdSense).
- Estar informados rápida y eficazmente de lo que les interesa y eligen (sindicación).
- Localizar la información más fiable.
- Poder colaborar o participar sin tener que rehacer (servicios Web).
- Poder disfrutar, por sobre todo de una buena interfaz de cliente.

11. Referencias

- AdSense de Google: Publicidad Inteligente – último acceso: 29/06/09
  http://www.maestrosdelweb.com/editorial/adsense/
- Aplicaciones de Web 2.0 – último acceso: 29/06/09
  http://www.documentalistaenredado.net/452/la-web-20-y-algunas-de-aplicaciones/
- Blogs – último acceso: 29/06/09
  www.maestrosdelweb.com/editorial/blogs/
- CMS: Sistema de Gestión de Contenidos
  http://es.wikipedia.org/wiki/Sistema_de_gesti%C3%B3n_de_contenido
- Ejemplos de Google AdSense – último acceso: 29/06/09
  http://www.google.com/adsense
- Flickr – último acceso: 29/06/09
  http://www.maestrosdelweb.com/editorial/flickr/
- Folcsonomía – último acceso: 29/06/09
  http://es.wikipedia.org/wiki/Folcsonomía
- Folcsonomías y Tags – último acceso: 29/06/09
  http://www.masternewmedia.org/es/2006/02/02/
  metodos_de_clasificacion_folcsonomia_y.htm
- Web 2.0 – último acceso: 29/06/09
  http://www.maestrosdelweb.com/editorial/web2/
- Web 2.0 Inteligencia colectiva – último acceso: 29/06/09
  http://www.planetaweb2.net/
- Web 2.0. Patrones de diseño y modelos del negocio – último acceso: 29/06/09
  http://sociedaddelainformacion.telefonica.es/jsp/articulos/detalle.jsp?elem=2146
- Web 2.0 y Web 3.0 – último acceso: 29/06/09
  http://www.lawebsemantica.com/contents/web20/web2.0-1.html
- Web Educativa 2.0 – último acceso: 29/06/09
  http://www.uib.es/depart/gte/gte/edutec-e/revelec20/anibal20.htm
- Wiki – último acceso: 29/06/09
  http://www.maestrosdelweb.com/editorial/queeswiki/

3. Funcionalidades Web 2.0 de Genexus

1. Introducción

Artech se encuentra en proceso de lanzar al mercado una nueva versión de Genexus, Genexus Rocha. Actualmente se encuentra disponible una versión RC (release candidate) la cual hemos usado para pruebas.

Se promete para esta nueva versión por parte de sus creadores, una mejora en la calidad de aquellas aplicaciones generadas para Web 2.0.

2. Genexus Rocha

Genexus Rocha es una herramienta con un entorno de desarrollo (IDE) gráfico.

En el IDE se puede encontrar una ventana principal que corresponde al objeto con el que se está trabajando. Por ejemplo, en el caso de que se este creando una página web el objeto corresponde a un Web Panel.

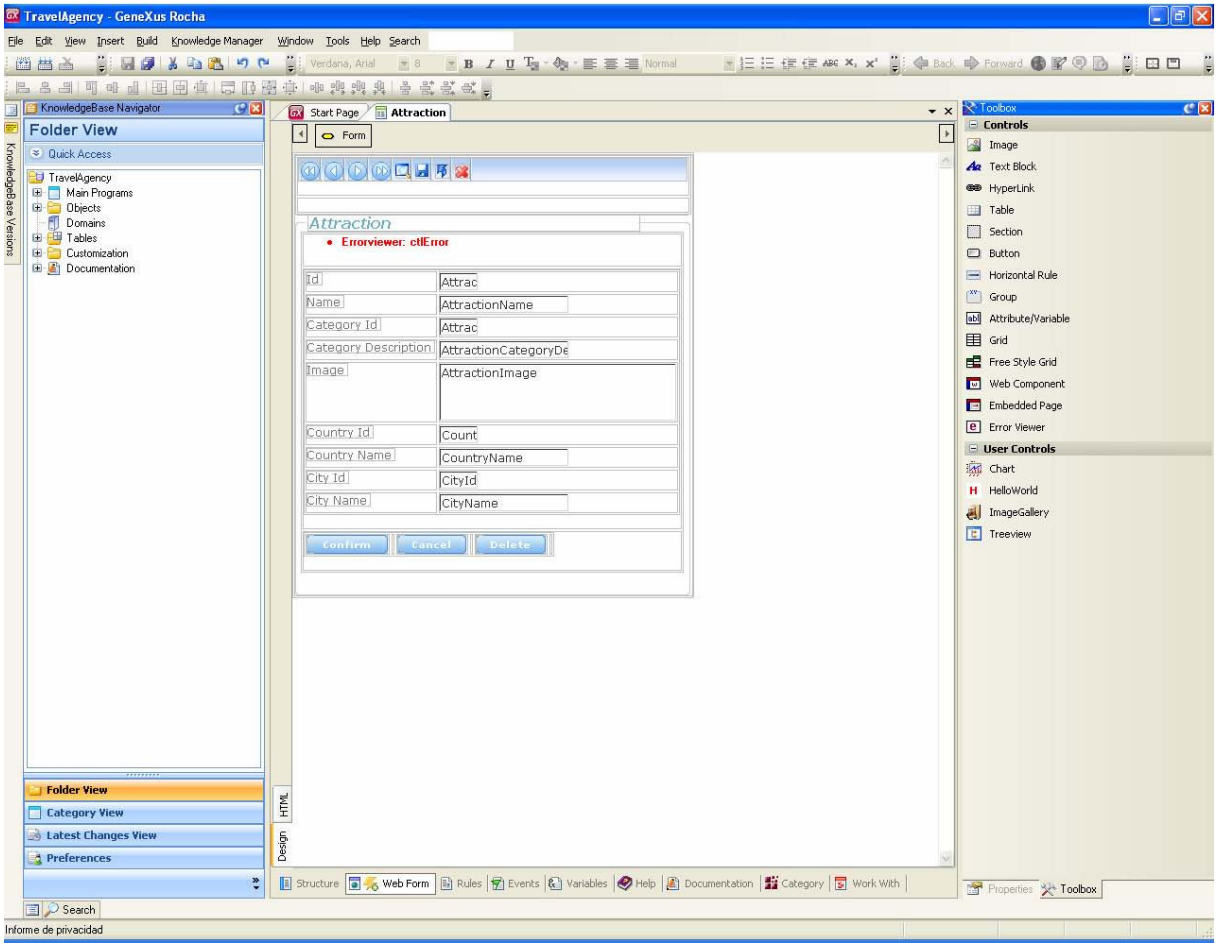


Figura 1: Entorno de desarrollo gráfico.

Se cuenta con varias ventanas secundarias y barras de herramientas. Por ejemplo, a la izquierda de la imagen se tiene El Navegador que es una ventana con categorías y jerarquías a partir de la cual se puede acceder a los distintos objetos contenidos en un proyecto (llamado en Genexus Base de Conocimiento). Como ejemplo de ventana secundaria se tiene la de Propiedades, que permite acceder a las propiedades del objeto que se encuentra en edición o

seleccionado. También se puede tener la ventana de Herramientas, en la imagen a la derecha, que permite agregar los diferentes controles que componen una página web, en este caso.

Entendemos por controles a objetos que están disponibles en el entorno de desarrollo, que tienen propiedades que pueden ser configuradas para obtener el comportamiento deseado y que también tienen eventos que se ejecutan y que permiten programar la lógica adecuada a su funcionamiento. Algunos de ellos son: bloques de texto, botones, imágenes, links, tablas, grillas. Como ejemplo de propiedades, para las grillas se permite indicar cantidad y nombres de columnas, su tamaño, su posición dentro de una página, colores, etc. Un evento que tiene disponible es el que ocurre cuando un usuario hace click en la fila de una grilla, o en un link contenido en alguna de sus celdas. Estos controles vienen definidos por defecto en Genexus.

3. User Controls

3.1 Generalidad

Genexus Rocha ofrece inicialmente un conjunto de User Controls ya desarrollados. Los User Controls son controles más complejos que los controles estándar, desarrollados por los usuarios. Ellos permiten enriquecer las interfaces.

Además, Genexus Rocha permite que los programadores puedan crear nuevos controles permitiendo generar mejores interfaces, crear controles que no hayan sido provistos por Genexus o controles que sean contenedores de controles que ya existen, controles que se adecuen a necesidades particulares de los usuarios. Esto es posible a través de la herramienta UserControlEditor que se instala bajo las carpetas de instalación de Genexus. Luego, estos nuevos User Controls pueden ser instalados e inmediatamente pueden comenzar a ser usados desde el IDE de Genexus como cualquier otro control predefinido por Genexus.

3.2 Controles ya definidos

Genexus cuenta con un catálogo de User Controls ya definidos o en procesos de definición. La comunidad de Genexus está compartiendo código abierto, estos incluyen muchos de estos controles. Se hace a través de la página www.gxopen.com. Al ser código abierto existe la posibilidad de modificar el funcionamiento de alguno de ellos a conveniencia o de agregarles más funcionalidad.

Se describen algunos de estos controles que resultaron de interés.

GxChart Control

Permite visualizar gráficas creadas dinámicamente.

Luego de agregado un control se establecen las categorías a representar en la gráfica y sus valores. Algunas propiedades que pueden definirse son títulos, leyendas, colores de los gráficos, valores para la escala y si se muestran valores o no.

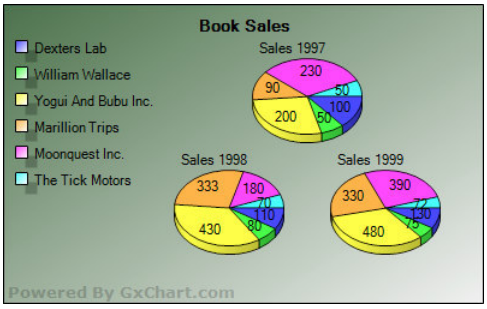


Figura 2: Control para gráficas.

ImageGallery Control

Este control permite mostrar e interactuar con imágenes al usuario. Se debe especificar una colección de imágenes a ser mostradas en el control. Se puede especificar textos a ser desplegados cuando la imagen se muestra.

TreeView Control

Permite visualizar información en forma jerárquica. Puede ser personalizado en tiempo de diseño y también en tiempo de ejecución. Permite que cada nodo pueda tener un icono diferente asignado. Permite que se programe la lógica adecuada para cuando el usuario selecciona uno de los nodos. También, cargar el árbol de manera dinámica.



Figura 3: Control para información jerárquica.

SlideMenu Control

Es un menú que puede ser agregado a una página y que permite mantener categorías de links. Cada ítem puede expandirse o contraerse en tiempos que se establecen como propiedad del control. Existen varios estilos de controles que permiten desplegar menús.

Map

Permite trabajar con mapas. Se puede establecer precisiones de la imagen, establecer latitud y longitud, usar zoom, marcar puntos en el mapa e incluir para estos un comentario o link. Existen para este control 3 proveedores, Google y Yahoo a nivel mundial e ICA (Ingenieros Consultores Asociados) para el departamento de Montevideo.

HoverPanel Control

Dada una imagen o un bloque de texto en una página web, este control permite que cuando el usuario pasa el cursor sobre ella, se muestre otro panel con información detallada acerca de la imagen o el texto.

Para poder usar este control, debido a cómo está implementado, debe definirse información en una transacción (una clase de objeto en Genexus) y contar con 2 Web Panel.

Tal como está implementado hasta el momento, este control sólo admite imágenes o bloques de texto. Este es un ejemplo en el cual se podría seguir agregando funcionalidad para que no sólo permita trabajar con imágenes o bloques de texto, sino que también con otros controles como podría ser celdas de una grilla.



Figura 4: Control para imágenes y texto.

Dolphin Style Menu

Este control es un menú horizontal con tabs. Permite asignarle a cada ítem del menú un título, descripción y un link a donde se direccionará cuando el usuario haga click.

Silverlight Photo Gallery Control

Muestra imágenes en una forma elegante y permite interactuar con ellas, está basada en tecnología Silverlight y maneja eventos Ajax para permitir trabajar con imágenes.

3.3 Creación de nuevos controles

Para crear los controles Genexus provee la herramienta, también gráfica, UserControlEditor. Este programa se instala bajo las carpetas de instalación de Genexus.

La definición de un control se divide en 3 grupos: General, Tiempo de Diseño y Tiempo de Ejecución.

Básicamente, en General se debe definir:

- El nombre del control.
- El nombre del archivo XML en el cual se guardará la definición de las propiedades del control. Estas propiedades son las que permiten configurar el control desde el IDE cuando se está desarrollando. Algunas podrían ser alcance (si está disponible en tiempo de diseño o en tiempo de ejecución), el dominio de la propiedad, el control en que se permitirá ingresar el valor del dominio, el valor por defecto.
- Indicar si el control estará disponible en la barra de herramientas o si sólo será un control de información.
- También indicar si es un control al cual se le pueda cambiar el tamaño en tiempo de diseño o si su tamaño es fijo.

En la siguiente imagen se puede ver el tab de Propiedades de la herramienta. Se ve que la definición de las propiedades se hace en forma gráfica. Luego se guardará en formato XML, como se mencionó en el párrafo anterior.

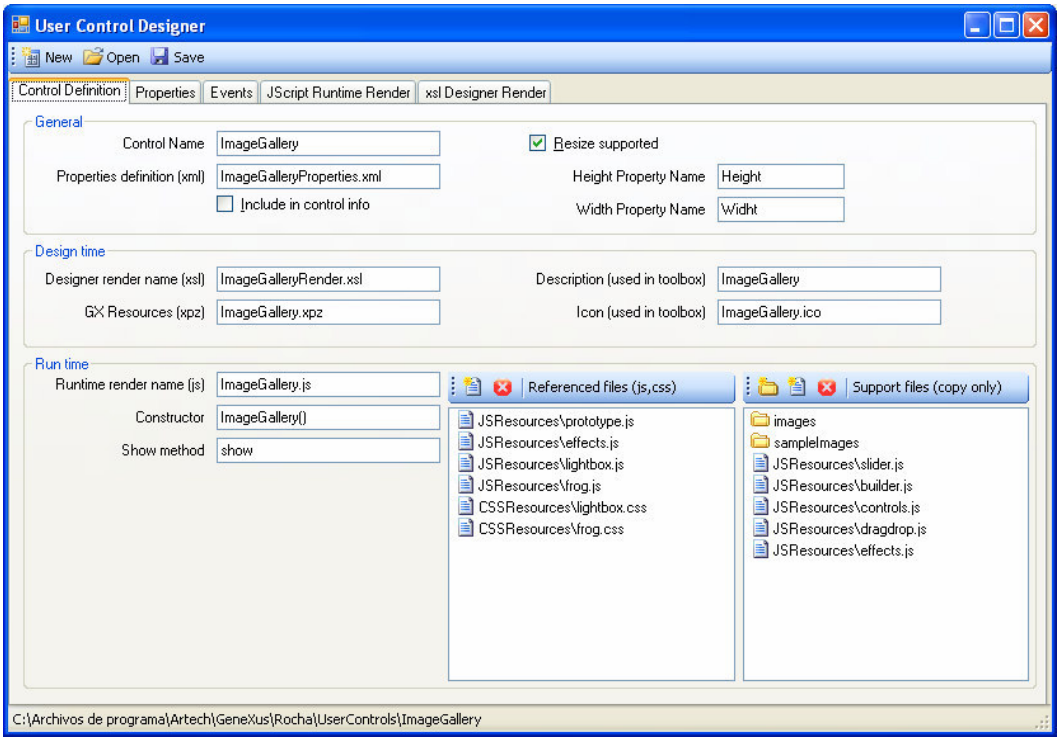


Figura 5: Crear control, definición.

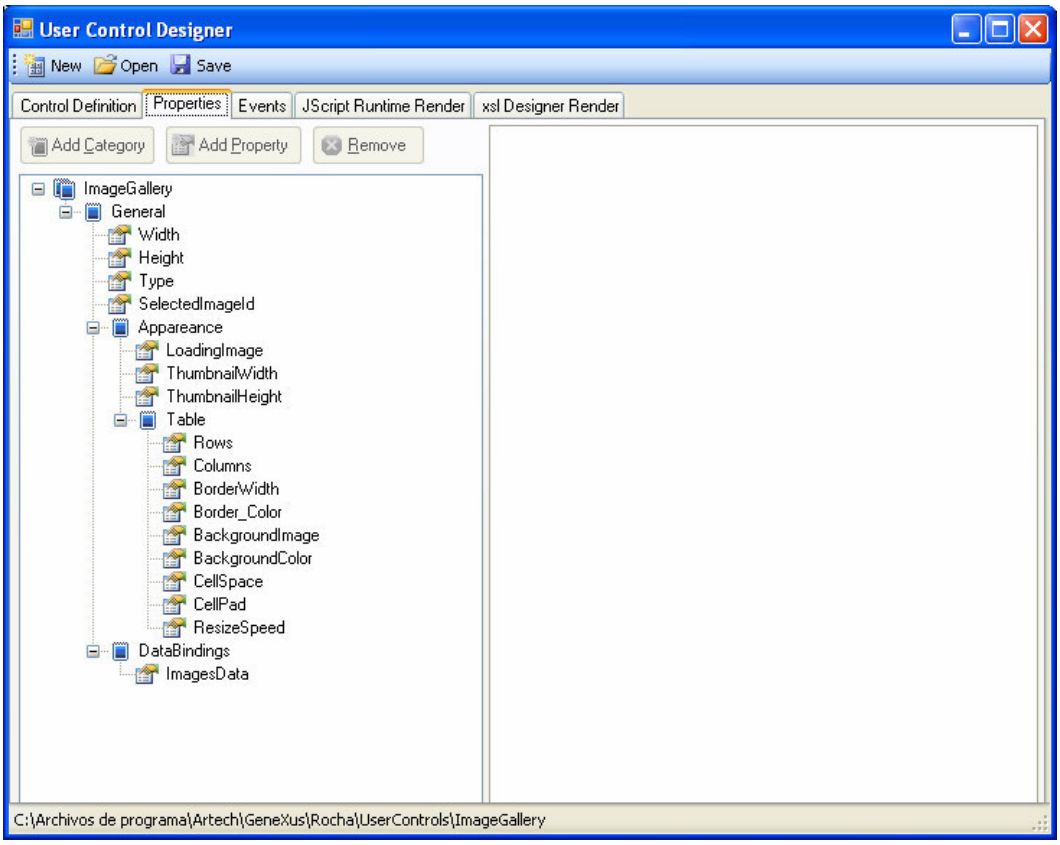


Figura 6: Crear control, propiedades.

En el grupo de Tiempo de Diseño se debe definir:

- El nombre del archivo xsl de definición de diseño, el cual define cómo se mostrará el control en el IDE.
- El nombre del archivo xpz que define algunos recursos extras. Esto sirve para el caso en que el control este asociado a datos, ya sea una variable, un atributo o un tipo de dato estructurado. Al ser incluido en el control, cuando el programador quiera definir una nueva instancia de este tipo podrá hacerlo sin tener que definir nada extra, ya que Genexus habrá consolidado la información aquí especificada.
- El icono que se desplegará en la barra de herramientas para representar a este control.
- La descripción del control que se despliega en la barra de herramientas.

En la siguiente imagen se puede ver el tab de Definición del archivo xsl.

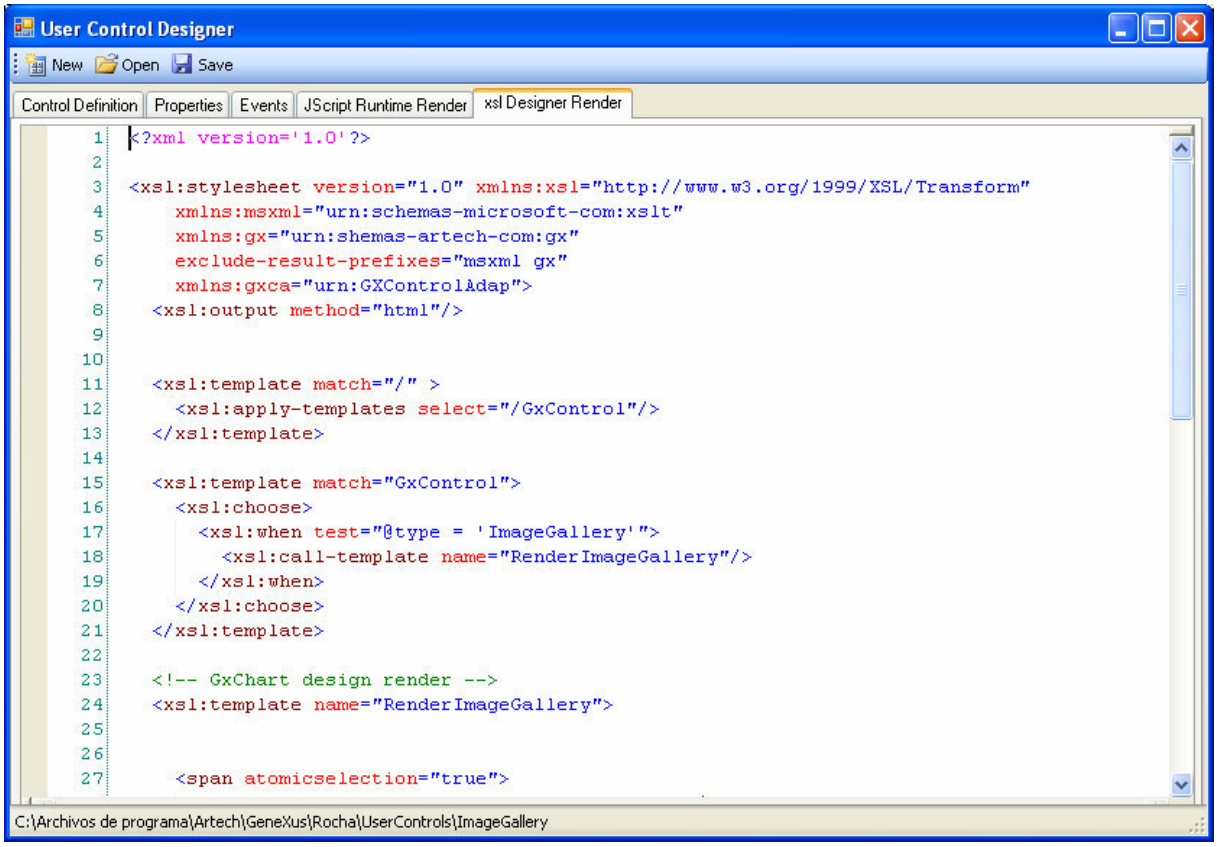


Figura 7: Crear control, diseño.

Para el grupo de Tiempo de Ejecución se debe definir:

- El nombre del archivo js que contiene la lógica para desplegar el control en tiempo de ejecución. Inicialmente, la herramienta crea el esqueleto para el script y el programador debe definir los Get y Set de cada método.
- El nombre del constructor para el archivo js. Genexus lo necesita para poder crear una instancia de la clase que queda definida por el mismo.
- El nombre para el método show. Este método es invocado luego de que se crea la instancia para mostrar el control.
- Archivos de referencia asociados y necesarios para crear el control. Se usan en caso de que se usen librerías externas (java script o archivos css).
- Archivos de soporte usados por el control pero que no son archivos de referencia. Estos pueden ser, por ejemplo, imágenes que se muestran luego en el control, como pueden ser nodos de un árbol.

La siguiente imagen corresponde al tab donde se define el archivo js.

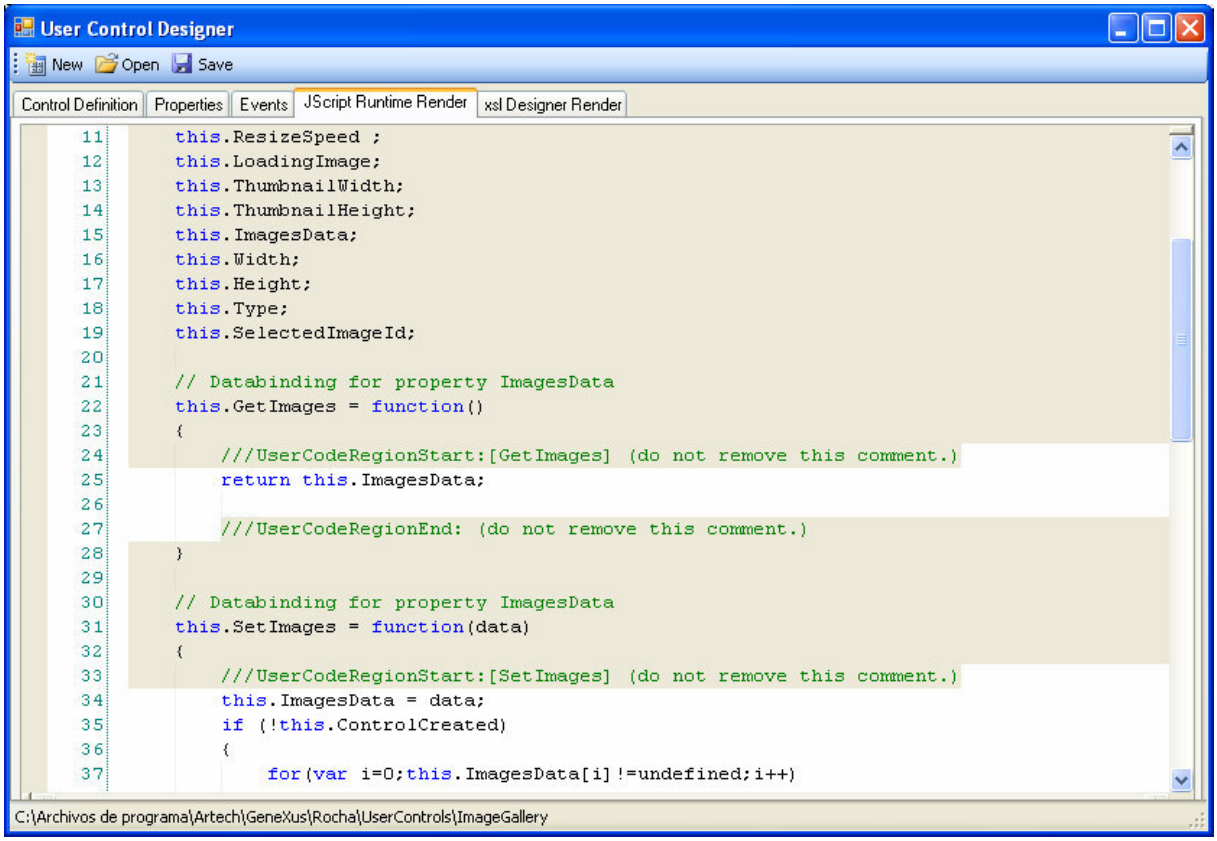


Figura 8: Crear control, ejecución.

Los User Controls tienen definida una carpeta en donde deben ser ubicados los archivos que se obtienen luego de la creación de los mismos. Esta carpeta corresponde a ..\Archivos de programa\Artech\GeneXus\Rocha\UserControls. Una vez ubicados los archivos del control en esta carpeta, solamente debe ejecutarse el comando genexus.exe/install para que el nuevo control quede instalado.

Luego de esto, el control aparecerá disponible en la barra de herramientas de Genexus y listo para poder ser usado.

4. Próxima versión

En la próxima versión de este documento se pretende agregar información sobre un proyecto existente dentro de los proyectos de código abierto de la comunidad Genexus. El mismo recibe el nombre de GxWiki y ofrece la posibilidad de implementar una Wiki. Este proyecto ha sido creado y está siendo modificado con Genexus Rocha.

Lo que se expone como código abierto es un 'kernel' a partir del cual se pueden realizar modificaciones para extender su funcionamiento. Los encargados del mantenimiento del kernel son ARTech y GXOpen Task Force. Existe la posibilidad de comunicarse con ellos en casos concernientes al mantenimiento del 'kernel' o temas asociados al proyecto.

Además de este proyecto, existen muchos más. La idea es incluir en la próxima versión algún otro proyecto que pudiera resultar de interés para nuestro proyecto.

5. Referencias

- Acceso a catálogo de User Controls ya definidos o en período de definición – último acceso: 29/06/09
  http://wiki.gxtechnical.com/commwiki/servlet/hwiki?category%3AUser+Controls
- Ejemplo de la Herramienta User Control Editor – último acceso: 29/06/09
  http://wiki.gxtechnical.com/commwiki/servlet/hwiki?User+Controls+-+Full+Documentation
- Ejemplo del User Controls Hello World. Es un control básico a modo de ejemplo. Se permite visualizar la interfaz de la herramienta UserControlEditor y el uso del control desde el IDE de Genexus – último acceso: 29/06/09
  http://wiki.gxtechnical.com/commwiki/servlet/hwiki?Hello+World+User+Control
- Proyecto GxWiki – último acceso: 29/06/09
  http://wiki.gxtechnical.com/commwiki/servlet/hwiki?Lanzamiento+de+GXwiki

4. Estudio de iTunes U

1. Introducción

En el mes de marzo de 2007 la compañía Apple lanzó el producto iTunes U (iTunes University) una nueva sección de iTunes store (tienda on-line de iTunes) en la cual se puede disponer en forma gratuita de una amplia gama de materiales y contenidos referentes a educación y cultura proveídos por las más afamadas y prestigiosas universidades de Estados Unidos y Canadá.

La causa que motiva la existencia de este tipo de herramientas es la de apoyar las principales metas de la educación a través de la innovación tecnológica continua. Se potencian así los sistemas de enseñanza, aprendizaje e investigación en la que tanto estudiantes como profesores e investigadores puedan disponer de un ambiente que colme sus expectativas y que acompañe su actual estilo de vida digital.

Esta nueva posibilidad que brinda Internet proporciona una manera creativa e innovadora para que millones de personas en el mundo entero puedan acceder en todo momento a lecturas, seminarios, clases, contenidos académicos y todo tipo de material educativo, así como también a la experiencia intelectual en investigación y descubrimiento que cada universidad aporta a la comunidad.

iTunes U es una herramienta de distribución simple y portable de fácil utilización. Creada para ofrecer un ágil intercambio de información educativa en archivos de audio, video y formato .pdf, iTunes U funciona como un espacio de almacenamiento administrado por Apple. Las personas que deseen utilizarlo solo tienen que descargar e instalar una aplicación gratuita llamada iTunes de Internet en su computadora a través de la cual podrán administrar y reproducir los archivos de interés. Por ser un producto de Apple, el iPod también resulta una forma de reproducción y traslado de estos archivos, haciendo que aprender una clase sea tan fácil como escuchar música.

iTunes U brinda principalmente la capacidad de buscar los contenidos públicos en la iTunes store de las instituciones participantes, además de permitir la posibilidad de proteger los álbumes de cursos para intercambio educacional con contraseña, organizando todo el contenido ofrecido en secciones para facilitar la búsqueda de material por tema o asunto y permitiendo la opción de configurar un podcasting directamente desde los álbumes. Los estudiantes, eventualmente, pueden disponer de una funcionalidad de buzones que les permite entregar tareas domiciliarias y la posibilidad de compartir materiales originales con otros miembros del curso.

2. Requerimientos de sistema

Desde el punto de vista de la institución educativa que desea poner a disposición sus contenidos digitales a la comunidad, no es necesaria la incorporación de nuevas tecnologías o sistemas, dado que es posible brindar acceso a iTunes U simplemente agregando esta funcionalidad a la infraestructura web ya existente. Además de estar especialmente diseñado para integrarse con ajustes mínimos a los sistemas de autenticación y autorización web implementados, es de fácil e intuitiva administración por lo que no implica recursos extra en este aspecto.

Para el público general (potenciales consumidores de contenidos) solo es necesario disponer de una conexión a Internet, una versión actualizada de la aplicación iTunes y, eventualmente, la posibilidad de poder visualizar los archivos descargados. Estos elementos son encontrados cada vez con más frecuencia en los hogares actualmente.

3. Organización y acceso de la información

Existen diferentes maneras de colocar los contenidos a disposición de los usuarios y que estos puedan acceder a los mismos.

Ciertos contenidos pueden ser publicados en iTunes store y, por consiguiente, estar accesibles para el público en general, mientras que otros, por su característica, pueden estar accesibles exclusivamente para miembros de una institución educacional particular participante del proyecto, debiendo los usuarios estar vinculados a la misma para presentar o consumir material en los diferentes cursos.

Las diferentes universidades pueden disponer de espacios de acceso restringido exclusivo a los miembros de su comunidad para todos sus cursos académicos, donde pueden encontrar una variedad importante de contenidos disponibles y/o sencillamente pueden presentar sus contenidos al público en general por medio de iTunes store.

Los contenidos educativos de cada institución son organizados y presentados en páginas que pueden corresponder a cursos específicos, áreas generales, áreas de institutos, o incluso planes enteros y donde el estilo depende de cual sea la institución participante.

Los usuarios de una institución académica pueden acceder a iTunes U simplemente entrando a la página web dedicada para este fin. Allí deben optar por ver el material público disponible accediendo directamente a iTunes store o identificarse logueándose para poder ver álbumes de cursos u otros contenidos exclusivos para miembros de dicha comunidad.

Una vez seleccionado el tipo de material de interés, el usuario puede ejecutar la aplicación iTunes y de esta forma ver o escuchar on-line, si así lo prefiere, los contenidos en cuestión o descargarlos a una librería en el disco local de su computadora para poder acceder a ellos más tarde sin necesidad de tener que ingresar nuevamente a iTunes U.

4. iTunes U como tecnología educacional

Como herramienta de distribución para material de audio y video digital seleccionado por el profesor. Los profesores pueden colocar material y organizarlo en pestañas que se corresponden con categorías llamadas tabs para que los estudiantes puedan encontrar fácilmente el material que necesitan.

Los estudiantes pueden descargar todo el contenido de un tab o hacerlo de a un archivo por vez.

Como un espacio para que los estudiantes compartan sus creaciones. Los profesores pueden establecer un tab como compartido para que los estudiantes puedan postear sus archivos y toda la clase tenga acceso a ellos.

Como un buzón de entrega de deberes. Los profesores pueden configurar la función de una tab como buzón para que los estudiantes suban sus tareas domiciliarias. El profesor es la única persona que tiene acceso al contenido del buzón de forma tal de mantener la confidencialidad de los trabajos.

Como pantalla de presentación para contenidos académicos. Los profesores en conjunto con coordinadores de carrera pueden publicar las presentaciones académicas en iTunes U de esa forma difundir los trabajos disponibles para esa comunidad universitaria o para el público general.

5. Caso de uso

Para que una entidad de educación pueda incorporarse al proyecto iTunes U es necesario un previo acuerdo con Apple. Una vez aceptadas las partes, Apple crea un nuevo sitio donde publicar o difundir los contenidos y actividades de dicha institución.

Luego de verificar que los datos enviados son correctos, se debe activar el sitio de forma tal de quedar habilitado para comenzar a planearlo, configurarlo y personalizarlo.

Es necesario además preparar una script que integre el ingreso de usuarios al sitio iTunes U con el sistema de identificación que ya se maneje en la institución.

Al intentar entrar al sitio, dependiendo de las especificaciones dadas, puede presentarse una página de bienvenida (normalmente reservada para contenidos no públicos) o una de características publicas.

Como ya se mencionó, se pueden definir páginas de cursos que contengan las creaciones de estudiantes o profesores y organizar dicho contenido en grupos, asignando permisos a cada grupo controlando de esta forma el acceso que los usuarios tienen al contenido. Así, los profesores pueden crear grupos para permitir entrega de revisiones de los estudiantes (grupo buzón) o compartir sus trabajos con otros participantes de la clase (grupos compartidos).

Es posible identificar dos roles bien marcados en las personas que forman parte del sistema. Podemos entonces, diferenciar el uso de este sistema desde el punto de vista institucional a través de quienes conforman una institución educativa que desean colocar y compartir información de interés, o desde la visión de aquellas personas usuarias, independientemente de su condición, que se dedican a consumir dichos contenidos.

5.1 Visión institucional

Planificación y organización

Cada sitio iTunes U tiene su propia organización dependiendo de las particularidades y requisitos que tenga la institución. Su estructura y contenidos están organizados en una jerarquía de archivos y directorios.

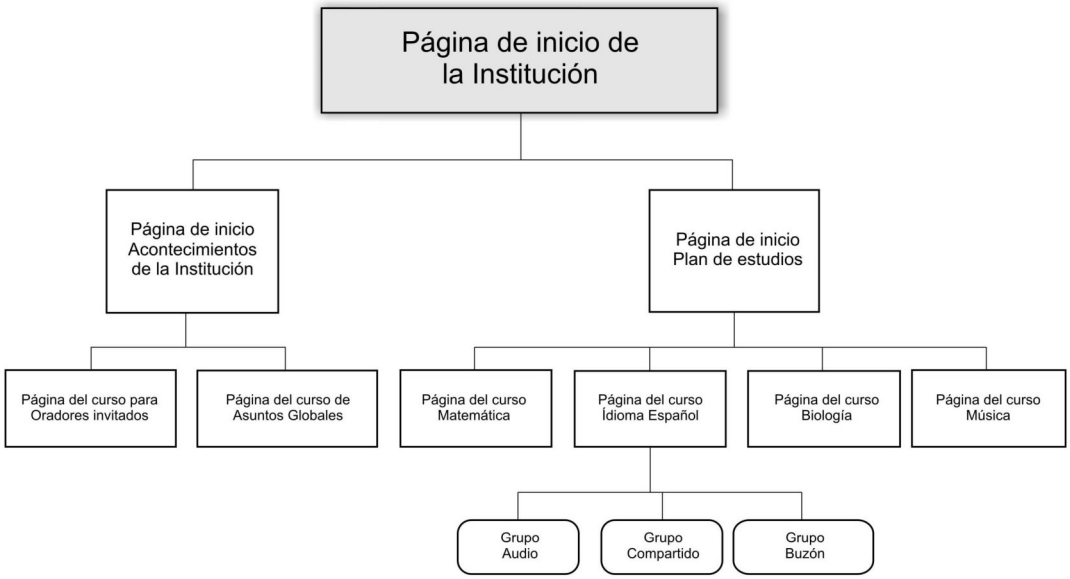


Figura 1: Organización jerárquica del sitio.

En este ejemplo es posible controlar el acceso de usuario proporcionando la entrada al sitio completo o restringir acceso a páginas de cursos o incluso descender en la estructura para dar permisos de acceso a grupos individuales dentro de un curso específico.

Para esto se cuenta con una clasificación de los diferentes usuarios en categorías definidas según su accesibilidad llamadas normalmente credenciales de usuario. Por defecto el sitio iTunes U proporciona credenciales predefinidas como pueden ser administrador, usuarios autenticados, usuarios no autenticados y todos. Es posible además, definir credenciales propias como ser de estudiantes o profesores que permite diferenciar los diferentes roles dentro de la institución de las personas autorizadas.

A cada usuario se le pueden definir diferentes niveles de acceso, que en combinación con las credenciales de usuario, definen la autorización, permisos y acciones que dicho usuario posee para las diferentes páginas, grupos o jerarquía de páginas.

El nivel de acceso asignado a la página inicial de la institución es heredado a las páginas y grupos del sitio, a menos que, específicamente se indique lo contrario. Por ejemplo, en el

diagrama anterior si a un usuario se le asignan credenciales y permisos para bajar contenido en la página inicial de eventos de la institución, por defecto recibe también dichos permisos para descargar de todas sus páginas descendientes (cursos para oradores invitados y cursos de asuntos globales).

Los contenidos educativos a compartir se presentan en las diferentes páginas y grupos definidos, que mediante la estructura antes descrita permiten el control de acceso a la información si así se requiere.

Servicios de autenticado y autorización

iTunes U asume que la institución educativa ya dispone de un sistema web que le provee de la infraestructura de autenticado y autorización de sus usuarios por lo que es necesario preparar una script que permita integrarse a dicho sistema, el script de transferencia.

De esta forma los administradores de la entidad pueden fácilmente mantener y controlar la información de usuario asegurándose el acceso confiable a los materiales de cursos disponibles en el sitio.

Para poder hacer efectiva esta integración es necesario modificar un ejemplo de script provista por el soporte técnico de iTunes U personalizando ciertos parámetros que son obtenidos inicialmente en el momento en que se activa el sitio.

Acceso autorizado

En el momento que una persona elige cliquear un enlace iTunes U en el sitio web de la institución se debe transferir a través de Secure Socket Layer (SSL) un string de autorización a iTunes U conteniendo las credenciales de usuario correspondientes y agregar opcionalmente información de identidad o firma digital.

Como ejemplo supongamos que el profesor Anthony Morrison intenta acceder al contenido del curso de sistemas operativos correspondiente al primer semestre del sitio universidad.edu, entonces el string señal de autorización es:

```
credentials=Instructor@urn:mace:universidad.edu:classes:1erSem:sistoper&
  identity="Anthony Morrison"<amorrison@universidad.edu>&time=1255487
```

Referencia a enlaces iTunes U

Cada página, grupo dentro de una página, así como un archivo dentro de un grupo en el sitio iTunes U de una universidad tienen una única URL permitiendo de esta manera una fácil referencia y acceso a la información compartida.

El siguiente diagrama ilustra lo sucedido cuando un usuario elige un link referenciando a cierto contenido iTunes U.

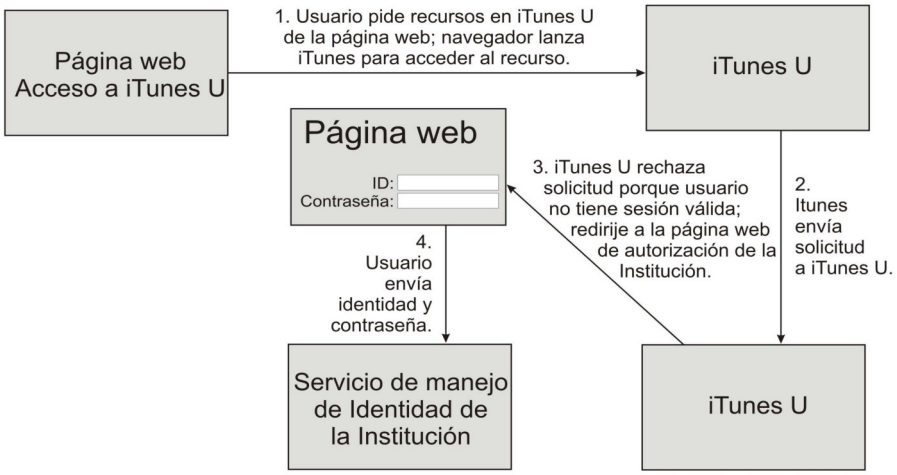


Figura 2: Eventos disparados al requerir contenido.

Incorporación de contenidos

A esta altura, resulta claro que el objetivo de una institución educativa que se une a iTunes U es el compartir archivos de audio, video y PDF para que tanto el estudiantado como el cuerpo de profesores e investigadores puedan colocar o descargar fácilmente tales contenidos aprovechando al máximo el avance de la tecnología aplicada a la educación. El software está especialmente diseñado para trabajar en conjunto con iTunes, iPod y iPhone, para manipular archivos de audio y video.

Una institución puede incorporar contenidos a su sitio iTunes U en forma manual o a través de Web service. Para agregar contenido al sitio manualmente se debe seguir una sencilla serie de pasos, similar a la de adjuntar un archivo en un correo electrónico. El proceso de subida de información a través de los Web Services implica la modificación al script de transferencia por parte del administrador del sistema. Con este cambio, se logra que dicho script genere una autorización que habilite la automatización del proceso de subida de contenidos.

Web Services de iTunes U

Los Web Services de iTunes U son un conjunto de operaciones definidas y descritas usando lenguaje XML. Es posible interactuar con los Web Services por medio de un servicio que posea el administrador de la institución o a través de la creación de nuevas herramientas, scripts o aplicaciones que podrán llamar remotamente los Web Services de iTunes U para automatizar muchas de las tareas administrativas.

Al crear aplicaciones o herramientas para interactuar con los Web Services se especifican operaciones, entidades e instrucciones iTunes U que le describen al servidor la forma en que se quieren ver o modificar la estructura y datos contenidos en el sitio iTunes U de la institución.

iTunes U provee un esquema de definición XML *(XML Schema Definition XSD)* para Web Services que en esencia tiene un propósito similar al de una definición de tipo documento *(Document Type Definition DTD)* y es usado para describir los elementos, atributos y restricciones de un documento XML.

Este XSD proporciona una descripción -basada en estándares-de la estructura de peticiones y respuestas a las transacciones con Web services de iTunes U.

5.2 Visión del usuario

Los profesores y administrativos pueden fácilmente colocar y cambiar contenido sin necesidad de apoyo de su departamento de sistemas.

Con iTunes U los usuarios pueden acceder a material educativo, incluyendo lecturas y entrevistas las 24 horas del día, 7 días a la semana donde pueden visualizarlo on-line o bajarlo al disco local de su computadora Mac o PC de forma de escuchar o ver el contenido más tarde o transferirlo a un iPod, iPhone o AppleTV para poder usarlo en cualquier lugar en cualquier momento. Los estudiantes pueden escuchar lecciones de lenguaje en el gimnasio, mirar demostraciones de laboratorio en un bar, revisar lecturas en el auto o en el ómnibus, permitiendo una flexibilidad importante en sus tiempos de estudio.

Requerimientos de sistema

iTunes U requiere iTunes versión 6.0 o posterior para Mac y Windows, Apple recomienda la versión 7.0 y para garantizar un mejor desempeño mantenerla siempre actualizada a su última versión.

También trabaja con los navegadores más modernos y estándar (Safari 2.0 o posterior para Mac, Internet Explorer 6.0 o posterior para Windows)

Por último, marcar que se debe contar con conexión a Internet para poder descargar los distintos contenidos.

Formatos de archivo soportados

Se permiten los formatos de archivos de audio y video comprimido, por lo que Apple recomienda:

- Archivos de audio (AAC, MP3) con las extensiones .m4a y .mp3 respectivamente.
- Archivos de video (MPEG-4 con compresión H.264) con sus correspondientes extensiones .m4v, .mp4 y .mov
- Portable document format (PDF)

Explorando iTunes U

iTunes U esta diseñado para proveer a estudiantes y profesores un fácil acceso a contenidos de cursos, así como también dar acceso a información institucional como eventos importantes, noticias y hasta mensajes del Decano u otras autoridades.

Dentro de las distintas universidades, cada curso tiene su propia página que inicialmente fue creada por el administrador del sitio.

Ejecutando la aplicación iTunes descargada gratuitamente desde el sitio de Apple Inc. y estando conectado a Internet es posible acceder al iTunes store que tiene una sección especial donde se presenta el iTunes U.

Es posible navegar dentro del iTunes U viendo el listado de las distintas universidades y colegios que participan del proyecto, pudiendo ingresar a cada una de ellas simplemente con seleccionarla y así desplegar la propuesta que estas presentan a la comunidad.

Una vez dentro del sitio de la entidad educativa se pueden navegar por las diferentes categorías en las que se clasifican todos los contenidos que la misma ofrece.



Figura 3: Portada de Universidad de Berkeley en iTunes U.

En la figura se puede apreciar una captura de la presentación del sitio iTunes U correspondiente a la universidad de Berkeley a través de iTunes donde es posible identificar la clasificación de contenidos que dicha universidad hace, separándolos en cursos, eventos, investigaciones, etc.

En el sector izquierdo se encuentra el panel de navegación por medio del cual es posible acceder tanto al iTunes store, como a la biblioteca local de contenidos multimedia descargados en el computador personal.

El acceso a los distintos artículos, clases, presentaciones y demás archivos de interés se hace una vez elegida la categoría que los contiene, donde con una interfaz sencilla invita a navegar, ver, suscribirse e incluso descargar dichos contenidos.

La imagen siguiente muestra cómo el usuario puede obtener pestañas (tabs) conteniendo listas de archivos con su correspondiente titulo, autor, duración, precio, etc., los cuales puede ver o escuchar en el momento o descargar a su computadora para acceder luego.

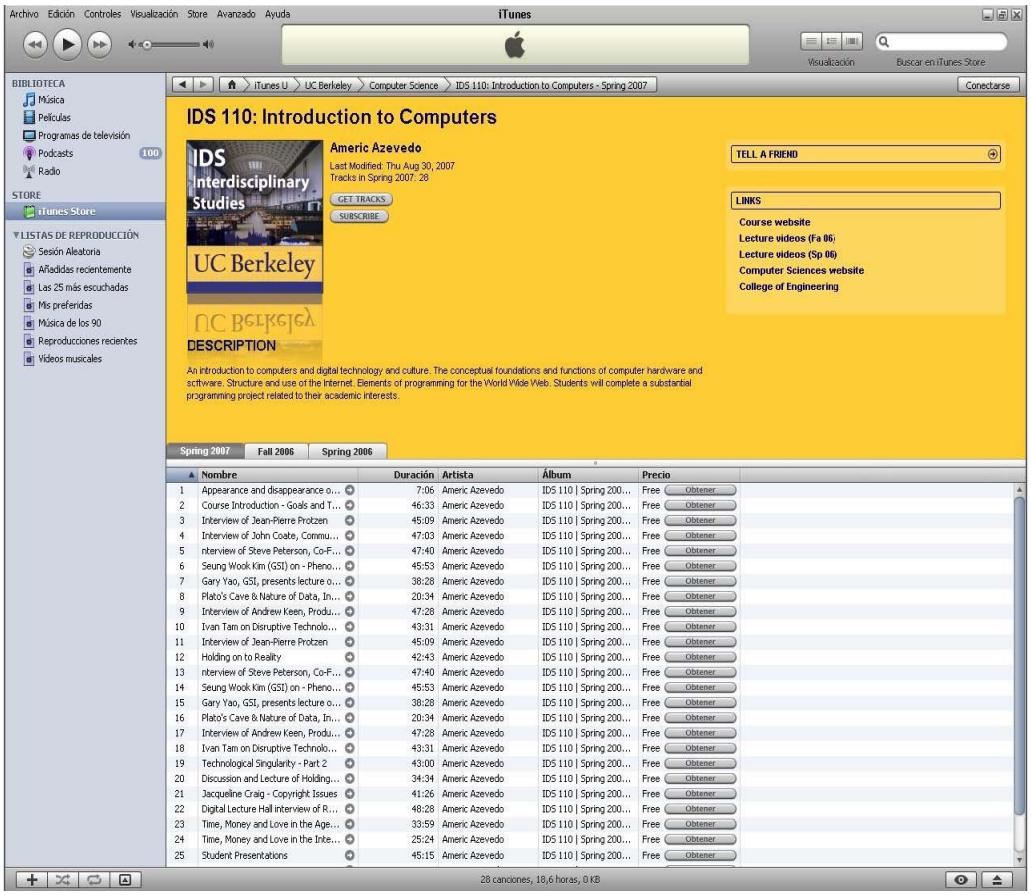


Figura 4: Listado de contenido disponible.

Fácil de usar, práctico e intuitivo, este sistema es una red que, aprovechando los avances tecnológicos y el alto impacto de la era digital en la sociedad actual, permite potenciar en gran forma la dinámica e interacción que las personas tienen frente a la educación continua.

6. Referencias

- Accessing iTunes University – último acceso: 29/06/09 http://www.youtube.com/watch?v=yZlRrMGnYj0&feature=related
- Apple education – iTunes administrator's guide – último acceso: 29/06/09 http://deimos.apple.com/rsrc/doc/iTunesUAdministratorsGuide/iTunesUAdministratorsGuide.pdf
- iTunes U – último acceso: 29/06/09 http://www.apple.com/support/itunes_u/
- iTunes U and Mobile learning – último acceso: 29/06/09 http://www.apple.com/education/itunesu_mobilelearning/itunesu.html
- iTunes U – Registred users – último acceso: 29/06/09 http://www.youtube.com/watch?v=efN6J3nM0jg
- iTunes U Step 1: Initiation – último acceso: 29/06/09 http://www.youtube.com/watch?v=NL_XUHwCe9o&feature=related
- iTunes U Step 2: Uploading – último acceso: 29/06/09 http://www.youtube.com/watch?v=OCiSJn2H5Tk&feature=related
- iTunes U Step 3: Consuming – último acceso: 29/06/09 http://www.youtube.com/watch?v=FNMDUZCIh7E&feature=related
- iTunes U Step 4: Connecting – último acceso: 29/06/09

http://www.youtube.com/watch?v=AQMuBiPnAxk&feature=related

- Login to iTunes U as a Guest – último acceso: 29/06/09
  http://www.youtube.com/watch?v=Ir4-Zye2GoU&feature=related
- Recursos audiovisuales sobre el modo de uso de iTunes U (inglés) – último acceso: 29/06/09

  How to Subscribe to a Podcast Using UIS on iTunes U
  http://www.youtube.com/watch?v=WpSmiBnl4yM&feature=related

  How to Use iTunes: How to Use iTunes Universities Feature
  http://www.youtube.com/watch?v=peYDd4G5tU8&feature=related
- Welcome to iTunes U – último acceso: 29/06/09
  http://images.apple.com/support/itunes_u/docs/iTunesU_User_Guide_1S_acc.pdf

7. Apéndice La aplicación iTunes

iTunes es una aplicación para computadores personales diseñada por la empresa Apple Inc. con el fin de reproducir, organizar y comprar música. Es compatible con los sistemas operativos Mac OS X, Windows 2000, Windows XP o Windows Vista. Estadísticamente, en la actualidad es el reproductor más utilizado por los usuarios de iPod.

Los usuarios de iTunes tienen la posibilidad de organizar su música en listas de reproducción, editar la información de los archivos, grabar CDs, convertir archivos a diferentes formatos, comprar música por Internet a través de la tienda iTunes Store y activar un visualizador que presenta la música en forma gráfica.

Además del soportar listas de reproducción estáticas, iTunes soporta listas de reproducción inteligentes. Dichas listas son listas que pueden ser configuradas para que se actualicen automáticamente basada en una lista personalizada con criterio de selección. Diferentes criterios pueden ser ingresados para controlar diferentes aspectos de la lista.

iTunes lleva registros de las canciones al crear una biblioteca virtual, permitiendo a los usuarios acceder y editar las propiedades de las canciones. Estas propiedades, conocidas como metadatos, están guardadas en archivos locales.

Aunque no es capaz de mostrar archivos PDF, iTunes los lista dentro de la biblioteca, cuando estos se encuentran presentes dentro de alguna carpeta añadida, permitiendo abrirlos con el programa lector de archivos PDF que tenga el usuario.

Las canciones de la Biblioteca de iTunes pueden ser compartidas a través de la red local usando un protocolo, también creado por Apple, para éste propósito.



Figura 5: Vista particular de la aplicación iTunes.

Los usuarios pueden arrastrar cualquier archivo de video a la biblioteca de iTunes para catalogación y organización. Estos pueden ser vistos en un pequeño cuadro en la ventana principal de la aplicación, en una ventana aparte o en pantalla completa.

iTunes también soporta plugins de visualización y de dispositivos que pueden ser descargados gratuitamente de Apple. Los plugins de visualización permiten a los desarrolladores crear efectos visuales a partir de la música que se encuentra reproduciendo, mientras de los de dispositivos permiten el soporte de reproductores de música adicionales.

Otra de las bondades ofrecidas es el soporte interno para hacer podcasting. Función que permite a los usuarios suscribirse a podcasts en la tienda on-line o ingresando la URL del canal RSS directamente. Una vez suscrito, el podcast se descargará automáticamente y el usuario puede elegir actualizarlo semanalmente, diariamente, o manualmente.

iTunes puede automáticamente sincronizar música y videos de la biblioteca a un iPod cada vez que éste es conectado. La sincronización automática puede ser desactivada para que se permita copiar las canciones o listan en forma manual.

A2 – Anexos referidos a tecnologías y patrones de diseño

1. Ocho patrones básicos para el desarrollo de la Web 2.0

Los ocho patrones básicos detallados a continuación reflejan reglas a tener cuenta cuando se está frente a un emprendimiento que contiene algún componente web, son considerados 'Ingredientes del éxito en Web 2.0'.

1. Aprovechar la inteligencia Colectiva

Se trata de crear una arquitectura de participación que utiliza el 'efecto de red' éste se produce con la participación de muchos usuarios quienes tienen la posibilidad de agregar valor a la información, ya sea de forma directa, mediante sus participaciones en charlas, creando y subiendo contenido, compartiendo, recomendando información o en forma indirecta a través de sus búsquedas, navegación y compras.

Se consideran buenas prácticas, primero: brindar a los usuarios lo que están buscando, y en segundo lugar 'algo' que genere el efecto de red. Confiar en ellos, diseñar el software que se optimiza cuánto más usuarios lo utilizan e involucrarlos ya sea explícitamente o implícitamente. También proporcionar una marco amplio para la creación de contenidos ya sea personalizando la aplicación o facilitando la búsqueda de sus datos y usuarios.

Se debate: la calidad versus cantidad de aportes de usuarios, comunidades libre acceso o cerradas (detrás de un pago y /o contraseña), temas de responsabilidad social como lo son los temas legales, minoridad, xenofobia, pornografía, etc.

2. Los datos son el nuevo "Intel inside"

En Internet, el mayor valor está en los datos.

Antes, el valor estaba en la CPU, ahora enfocados en Internet el valor está en la calidad de los datos que se dispongan.

Existen algunas estrategias para darles valor, ellas son: su forma de crearlo, control, acceso e infraestructura.

En cuanto a su creación se enfatiza el hecho de ser propietario de datos difíciles y caros de regenerar, promover su creación a través del efecto de red.

Su control es enfocado en crear formatos personalizados, registros y directorios.

En cuanto a su acceso se intenta generar datos que no se encuentren en otros lugares y finalmente proveer infraestructura para almacenar y acceder a datos generados por 'otros'.

Se considera buenas prácticas:

- Intentar generar como propietario una base de datos única y difícil de recrear.
- Aplicar estrategias para agregar 'valor a los datos' y de esa manera mejorar la base, cuando se trata de datos que no son exclusivos ni difíciles de conseguir.
- Hacer que algunos derechos sean reservados pero no todos, tener cuidado cuando se agrega demasiada restricción ya que podría atentar contra el negocio.
- Definir la estrategia de flujo de datos, a qué compañía pasar qué datos.
- Ser propietario de índice, espacio de nombres, formatos (no de los datos), por ejemplo Google indexa las páginas pero no es su dueño.
- Diseñar la estructura de los datos de forma que puedan ser reutilizados, como es sabido es tan importante acceder a ellos como su reutilización, generando mecanismos de sindicación y distribución, contemplar mecanismo de acceso a datos, su diseño, presentación y licenciamiento.
- Tercerizar su administración o administrar datos de otros.

Se debate definir quien es su dueño, se habla de movimiento pro derechos de autor, movimiento de datos abiertos.

Se recomienda a organizaciones hacerlos reutilizables, sucede que datos corporativos, empresariales, organizaciones públicas se encuentran atrapados en base de datos y de aplicaciones porque las empresas no prevén su uso potencial.

Se les recomienda aprovechar formatos sencillos de distribución (por ejemplo RSS) para crear mecanismos de bajo costo que posibiliten su distribución interna y su uso.

3. Innovar en el ensamble

Enfocados en la era Web 2.0 se habla de protocolos de comunicación abiertos (HTML, XML, TCP/IP, HTTP), plataformas Web que proveen datos y servicios.

Alguno de sus beneficios son: proveer un modelo de crecimiento escalable, las APIs promueven la innovación de terceros, las plataformas abiertas desarrollan confianza y comunidad, los clientes nos muestran cómo son usados los servicios brindados, los modelos de ingresos se vinculan directamente a la plataforma.

Se considera buenas prácticas:

- Ofrecer como parte del servicio, APIs.
- Diseñar para remezclar por ejemplo hacer el contenido accesible en la unidad más pequeña posible como puede ser un mensaje.
- Aplicar las mejores prácticas de APIs: herramientas que favorecen el desarrollo como por ejemplo foros, blogs, mails.
- Utilizar estándares, como por ejemplo RSS y Atom para sindicar;
- Hacer que la API sea parte del modelo de negocio, si un tercero a través de ella generara una venta, ambos compartirían la ganancia.
- Direccionamiento granular del contenido, por ejemplo la unidad de los datos no es la página web, sino una foto, un mensaje individual de un post, etc.
- Aprender de los usos que la comunidad hace de las herramientas provistas, es decir si las aplicaciones desarrolladas por terceros (y que utilizan nuestros datos/APIs) prosperan, eso se debe reflejar en nuestra forma de hacer negocios.

Algunos de los temas de debate son: garantías sobre las APIs y atraso tecnológico.

Se recomienda a las organizaciones comenzar en pequeño, por ejemplo brindar cuestiones sencillas como sindicar los datos actualizados o vigilar el espacio del consumidor.

4. Experiencias ricas de usuario

Se apunta a evolucionar de lo que antes eran páginas estáticas, a aplicaciones de Internet. Se trabajaba con aplicaciones clientes 'pesadas', páginas estáticas en la web y ahora se apunta a desarrollar aplicaciones livianas que puedan correr sobre navegadores que permitan mayor interacción y sean independientes de la plataforma.

Toda esta evolución genera beneficios como por ejemplo: aumentar la tasa de satisfacción, menor abandono y mayores ventas, un mejor desempeño y ventajas competitivas.

Se considera buenas prácticas:

- Combinar lo mejor de las experiencias de escritorio y en línea: no se necesita instalar software en cada PC, se avanza en el trabajo colaborativo, ventajas en precio también.
- Primero usabilidad y simplicidad, luego funcionalidades: no es necesario tener todas las funcionalidades del MSWord para ser útil, es bueno facilitar la validación por terceros.

- Búsqueda sobre estructura: organizar la información a través de la búsqueda.
- Hacer que los contenidos estén asociados con direcciones: de esa forma no se obliga a que cada contenido tenga su dirección asociada.
- Personalización profunda y adaptativa: aprender del comportamiento del usuario, anticipar demandas por ejemplo comenzar a bajar un documento antes de definir dónde va a ser almacenado, recordar preferencias.

5. Software sobre el nivel de un único hardware

Se apunta a diseñar aplicaciones integradas, ya sea servicios entre PCs, celulares y servidores de Internet.

El PC ya no será el principal dispositivo de acceso a Internet, nos encontramos con problemas en plataformas en cuanto a la velocidad entrada/salida y limitaciones para funcionar sin conexión.

Se dice que el éxito de las aplicaciones dependerá en la facilidad de desarrollo que dispondrán en todo este escenario. Por ejemplo, las fotos pueden ser capturadas por celular, enviadas por mail o subidas a un servidor (Flickr), y con respecto al sonido se pueden bajar temas para ser escuchados en dispositivos portables o ser emitidos por streams a través de radios virtuales, entre varias posibilidades.

Algunas de las buenas prácticas a implantar es el diseñar desde el comienzo pensando en ser compartido a través de dispositivos, servidores y redes.

Pensar en navegar Web 2.0 desde un celular, utilizar por ejemplo también GPS incorporado en el dispositivo para determinar su ubicación.

Aprovechar la potencia de red para hacer más inteligentes a los clientes, se está frente a una situación en que los celulares son más potentes, pero los servidores también.

Hacer interfaces de un click pensando siempre en optimizar la habilidad de los clientes que usan nuestro servicio, como es sabido hay limitaciones con respecto a la interfaz que tienen algunos dispositivos como los celulares y las consola.

6. Beta perpetuo

Se está frente a una situación en que el software es un servicio al que hay que agregarle funcionalidades y es el usuario el encargo de testearlo.

Dejó de existir el ciclo: Diseño, Desarrollo, Testeo, Empaquetamiento, Transporte, Instalación, el usuario participa en todo momento. El software no se empaqueta más y el usuario queda en espera de actualizaciones continuas.

Todo esto es beneficioso en el sentido de minimizar el riesgo, generar mayor velocidad de respuestas, disminuir el tiempo de ingreso en el mercado y aumentar la relación con los clientes.

Se considera buenas prácticas la entrega temprana y frecuente: arreglos de Bugs y funcionalidades detectadas por el usuario.

Utilizar al usuario como tester: es mejor que cualquier documento de requerimiento prototipado, generalmente los usuarios realizan las acciones de manera diferente a como lo informan.

Crear productos nuevos de manera incremental, experimentar siempre con ideas nuevas, utilizar lenguaje independiente de la plataforma.

Se piensa que el test del usuario elimina la necesidad de garantizar la calidad, esto no es correcto, el usuario solo está para validar y mejorar las funcionalidades, no la calidad.

Se toma en cuenta la privacidad de la información, se debe establecer políticas de seguridad y privacidad sobre la información que se testea.

7. Aprovechar la larga cola

Se trata de aprovechar las ventajas de Internet para aprovechar nichos de mercados que antes eran pequeños, se habla de evolución: pasar de opciones limitadas y mercado masivo a opciones ilimitadas y nichos de mercados.

En Internet no existen almacenamiento limitado, no hay problemas de transporte ni de localización física, se puede dirigir un gran supermercado sin preocuparse de que todo deba estar concentrado en un solo lugar, se ahorra en costos debido a que no hay problemas de logísticas ni de intermediarios.

Se debe utilizar algoritmos para relacionar oferta y demanda y arquitectura de participación para relacionarla.

Se debate el hecho de que algunos mercados y bienes se benefician más de la larga cola por ejemplo los bienes digitales no tienen casi costos de almacenamiento y distribución.

Se piensa que los best Sellers no importan, eso no es correcto, sí importan puede considerarse como punto de partida (a través de recomendaciones y rankings) para ir hacia la cola, hacia intereses personales y especializados.

Para fomentarla se aconseja usar herramientas de integración basadas en wikis y mashup, así como también recuperar datos estructurados y no estructurados.

8. Modelos livianos y escalabilidad rentable

La escalabilidad se aplica a los modelos de negocios así como a las tecnologías.

La idea de todo esto es: 'hacer más con menos'. Hoy en día no es necesario que existan grandes equipos de desarrollo antes de que las aplicaciones se pongan en producción. Tampoco se requieren grandes presupuestos de marketing ya que con la disminución en cuanto a hardware y ancho de banda, como también la generación de publicidad alternativa (de boca en boca), ciclos iterativos de producción y mayor participación del cliente, reducen los costos, tiempo y riesgo.

Se apunta a escalar con la demanda, del lado de la demanda el efecto de red hace que el crecimiento sea exponencial, del lado de la oferta es más fácil ahora escalar tecnología y la relación ingresos/gastos.

Se espera crear negocios a partir de negocios de otros, agregando avisos, realizando transacciones electrónicas, búsquedas y chat.

Es una buena práctica también tercerizar siempre que sea práctico y posible, por ejemplo tercerizar almacenamiento (disco), desarrollo de software y test. Esto es llevado a cabo tanto por grandes como por pequeñas empresas.

Otra manera de tercerizar es saber que si uno es bueno haciendo algo u ofreciendo un servicio entonces ofrecerlo a otras empresas.

Utilizar avisos sindicados, con sponsors, por subscripción, pensar en diseñar para facilitar la escala, cuanto más simple más se escala.

El éxito de la Web 2.0 requiere innovación y éste requiere la toma de riesgos, hay que promover ideas nuevas pero también reconocer que los problemas y fallas ocurren.

Ha que tener en cuenta el efecto de red éste es bueno si se está bien parado para recibirlo en caso contrario hunde.

También hay que pensar que si se puede hacer un sitio barato y rápido seguramente también lo pueden hacer otros, no se puede depender solamente de la publicidad, tendrán mayores chances aquellos que apliquen principios de la Web 2.0.

9. Referencias

- Curso postgrado 'Web2.0 Aplicaciones Web avanzadas' de Universidad de la República, Facultad de Ingeniería, Instituto de Computación.

2. Modelo-Vista-Controlador

1. Concepto

Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario y la lógica de negocio en tres componentes distintos.

2. Objetivo

Su objetivo es distinguir entre la lógica de toda la aplicación y la presentación, evitando repetirse y enfocando a su organización.

3. Descripción

Es utilizado para aplicaciones con sofisticadas interfaces, la lógica de una interfaz de usuario cambia con más frecuencia que los almacenes de datos y la lógica de negocio.

Si se realiza un diseño que mezcle los componentes de interfaz y de negocio, entonces la consecuencia será que, cuando se necesite cambiar la interfaz se tendrá que modificar trabajosamente los componentes de negocio, lo que generaría mayor trabajo y más riesgo de error.

Se trata de realizar un diseño que independice la lógica de negocio de la interfaz gráfica y desacople la vista del modelo, con la finalidad de mejorar la reusabilidad.

De esta forma las modificaciones en las vistas impactan en menor medida en la lógica de negocio o de datos.

4. Elementos del Patrón

Se presenta a continuación un diagrama sencillo que muestra la relación entre el modelo, la vista y el controlador, las líneas sólidas indican una asociación directa, y las punteadas una indirecta

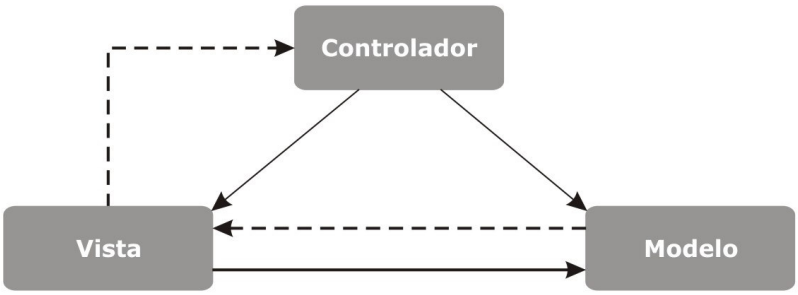


Figura 1: Diagrama del modelo.

4.1 Modelo

Es el responsable de acceso a datos, es la representación específica de la información con la cual el sistema opera.

La lógica de datos asegura la integridad de estos y permite derivar nuevos datos. Por ejemplo, no permite comprar un número de unidades negativo.

4.2 Vista

Despliega la información del modelo, en un formato adecuado para interactuar, más conocida como interfaz de usuario.

4.3 Controlador

Maneja el comportamiento del modelo ante eventos de la interfaz, normalmente son acciones que el usuario invoca, implica cambios en el modelo y también en la vista (interfaz).

5. Flujo de control

A continuación se despliega un ejemplo de Modelo-Vista-Controlador (MVC) para ilustrar cómo sería el flujo de control, es un ejemplo de navegación web se trata de un modelo pasivo (no notifica cambios en los datos) que responde a las entradas del usuario, pero no detecta los cambios en datos del servidor.

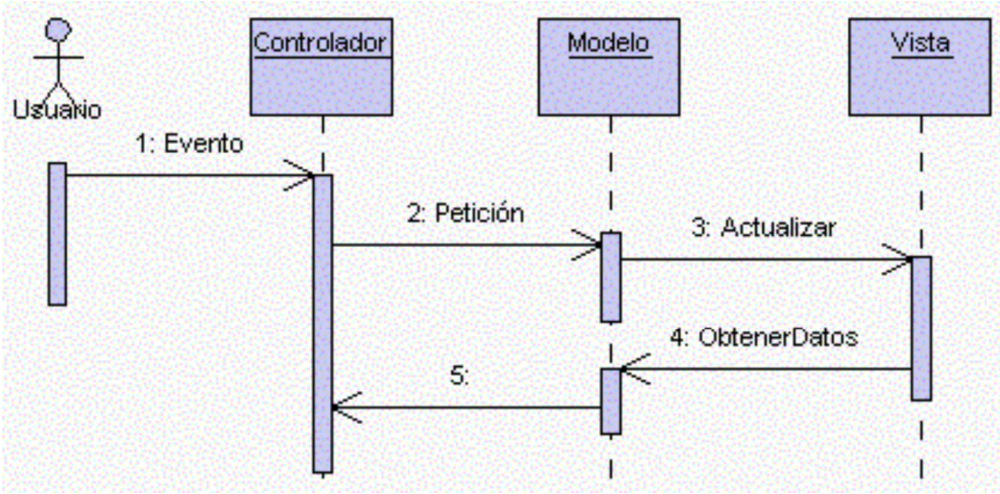


Figura 2: Flujo de control.

Pasos:

1- El usuario introduce el evento es decir interactúa con la interfaz.

2- El Controlador recibe el evento y lo traduce en una petición al Modelo, posiblemente impactando en su estado (aunque también puede llamar directamente a la vista).

3- El Modelo si es necesario llama a la Vista para su actualización.

4- Para cumplir con la actualización la Vista puede solicitar datos al Modelo.

5- El Controlador recibe el control.

6. Referencias

- Flujo de control – último acceso: 29/06/09
  http://www.proactiva-calidad.com/java/patrones/mvc.html
- Model View Controller – último acceso: 29/06/09
  http://msdn.microsoft.com/en-us/library/ms978748.aspx

3. Introducción a Windows Presentation Foundation (WPF)

Se trata de una tecnología proporcionada por Microsoft .NET Framework, lo que permite crear aplicaciones que incorporan otros elementos de la biblioteca de clases del framework. Ofrece una amplia infraestructura y potencialidad gráfica con la que se puede desarrollar aplicaciones de atractiva apariencia, con mayores y mejores facilidades de interacción que incluyen animación, video, audio, documentos, navegación y gráfica 3D.

Fue elegida como herramienta de desarrollo por presentar todo lo anteriormente descripto, permitiendo de esa manera con el uso de una única herramienta desarrollar todas las funcionalidades que la aplicación requería generando interfaces de usuarios ricas y eficientes, punto importante en la aplicación desarrollada.

1. Descripción

Como se sabe, el interés de los programadores se centra en el funcionamiento de las aplicaciones y no tanto en la forma de interacción que puede darse entre ellas y los usuarios. Sin embargo los usuarios dan gran importancia a la interfaz de la aplicación y constituye su experiencia. La experiencia mejorada de los usuarios mediante una interfaz optimizada puede contribuir al incremento de la productividad, al aumento de sus motivaciones para hacer uso de la herramienta, entre muchas otras ventajas.

Es importante destacar que quienes harán uso de *G-Sell* son usuarios con poca o casi ninguna interacción con una computadora, y que además no están en sus planes dominarla. Por lo que, la interfaz a desarrollar no solo tiene que 'atraparlos' para que continúen usándola sino que también debe ser ampliamente amigable, fácil de navegar, fácil de usar, hallar de manera rápida lo que buscan, etc. Todos esos puntos y más, son los que se han tenido en cuenta a la hora de dar mayor relevancia a sus interfaces.

2. Ventajas

Existen tres aspectos fundamentales de WPF, entre muchos otros que destacan su importancia, ellos son:

Plataforma unificada para interfaces de usuario modernas. Posibilidad de trabajar juntos desarrolladores y diseñadores. Tecnología común para interfaces de usuario de Windows y explorador web.

2.1 Plataforma unificada para interfaces de usuario modernas

Anteriormente la creación de una interfaz de usuario de Windows requería el uso de varias tecnologías diferentes, la introducción de WPF cambió este enfoque.

La siguiente tabla presenta un resumen de la situación.

	Windows Forms	PDF	Windows Forms/ GDI+	Windows Media Player	Direct 3D	WPF
Interfaz gráfica, como formularios y controles	X					X
Documentos en pantalla	X					X
Documentos de formato fijo		X				X
Imágenes				X		X

Vídeo y audio				X		X
Gráficos bidimensionales			X			X
Gráficos tridimensionales					X	X

Figura 1: Tabla plataforma unificada para interfaces de usuarios modernas.

Al proporcionar una amplia gama de funciones en una sola tecnología, WPF simplifica de forma significativa la creación de interfaces de usuario modernas.

De la lista desplegada *G-Sell* hizo uso de varios puntos de ellas. Si bien el enfoque primordial fue apuntar al diseño de una potente interfaz gráfica el uso de WPF facilitó, en sucesivos avances del desarrollo, la incorporación de nuevas funciones, como incluir publicidad (incorporación de banners), incluir información de publicaciones que son presentadas en forma de documentos, etc.

A continuación se ilustra un ejemplo de ello.



Figura 2: Inclusión de publicidad en la aplicación.

2.2 Posibilidad de trabajar juntos desarrolladores y diseñadores.

La colaboración entre desarrolladores y diseñadores se refleja mediante la generación de lenguaje de marcado de aplicaciones extensible (XAML) y código subyacente.

Primero se desarrollaron funcionalidades sin dar mucha importancia al diseño de las interfaces, luego que éstas se comportaran como se quería se afinó el diseño basado en el archivo xaml generado, logrando de esa manera las funcionalidades deseadas con una interfaz elaborada a medida.

Marcado

XAML es un lenguaje de marcado basado en XML que se utiliza para implementar la apariencia de una aplicación mediante declaración. Suele utilizarse para crear ventanas, cuadros de diálogo, páginas, controles de usuario, setear propiedades de los controles, formas y gráficos.

A continuación se despliega un ejemplo de XAML que implementa la apariencia de los tres botones (reproducir, pausar, detener) que se muestran en la pestaña de emisión por demanda de la aplicación. Éstos permitirán reproducir el video que se encuentra guardado localmente en el equipo del usuario, correspondiente a una entrevista ya emitida, así como también pausarlo cuando lo crea conveniente o detenerlo en el momento que desee.

```
<cc:ImageButton x:Name="Play"
       OriginalImage="/Integracion;component/imagenes/play.png"
       ToolTip="Comenzaremisión"
       MouseOverImage="/Integracion;component/imagenes/play
       apretado.png"  Width="42" HorizontalAlignment="Left"
       Margin="20.439,0,0,0"
       Grid.Column="1"     />
<cc:ImageButton x:Name="Pausa"
       OriginalImage="/Integracion;component/imagenes/pausa.png"
       ToolTip="Pausar emisión"
       MouseOverImage="/Integracion;component/imagenes/pausa
       apretado.png" Margin="67,0,31,0" Grid.Column="1" />
<cc:ImageButton x:Name="Stop"
       OriginalImage="/Integracion;component/imagenes/stop.png"
       ToolTip="Detener emisión"
       MouseOverImage="/Integracion;component/imagenes/stop
       apretado.png" Grid.Column="2" HorizontalAlignment="Left"
       Width="39.552" />
```

Código 1: Ejemplo XAML de botones en pestaña Emisiones por Demanda.

Mediante la propiedad ImageButton, correspondiente al control Button, XAML define un botón configurable con atributos, como el atributo ToolTip, el cual le otorga ayuda al usuario indicándole la función del botón. En tiempo de ejecución, WPF convierte los elementos y los atributos definidos en el marcado en instancias de clases de WPF.

A continuación se ilustra la interfaz de usuario definida mediante el código XAML del ejemplo anterior.



Figura 3: Interfaz de usuario.

Código subyacente

El comportamiento principal de una aplicación es implementar la funcionalidad que responde a las interacciones con el usuario, lo que incluye controlar los eventos (por ejemplo, hacer clic en un menú, una barra de herramientas o un botón) y llamar, en respuesta, a la lógica empresarial y de acceso a los datos. En WPF, este comportamiento se suele implementar en código asociado al marcado. Este tipo de código se denomina subyacente. En el ejemplo siguiente se muestran el marcado actualizado del ejemplo anterior y el código subyacente.

```
<cc:ImageButton x:Name="Play"
       OriginalImage="/Integracion;component/imagenes/play.png"
       ToolTip="Comenzaremisión"
       MouseOverImage="/Integracion;component/imagenes/play
       apretado.png"Click="Play_Click"Width="42"
       HorizontalAlignment="Left" Margin="20.439,0,0,0"
       Grid.Column="1" />
```

go

Códi
2:

Ejemplo XAML, evento del botón Play_Click.

El código de implementación descripto a continuación implementa un controlador de eventos para el evento Play_Click del botón. Cuando se hace clic en él, el controlador de eventos comienza la reproducción del video.

```
private void Play_Click(object sender, RoutedEventArgs e)
{
        ...

          th1 = new Thread(new ThreadStart(proceso1));
          th1.SetApartmentState(ApartmentState.STA);
          th1.Start();
          TransformGroup tg = new TransformGroup();
          RotateTransform rt = new RotateTransform(0);
          tg.Children.Add(rt);
          Reproductor.RenderTransform = tg;
          string fileUrl =  Transforma.FormatoURL("\\Emisiones\\",
               idEntrevista, Formato.Video);
          string Directorio =Path.GetDirectoryName
               (Assembly.GetExecutingAssembly().Location);
          string relativo = Directorio + fileUrl;
          Reproductor.Source = new Uri(relativo);
          Reproductor.Play();
          Play.IsEnabled = false;
          Pausa.IsEnabled = true;
           Stop.IsEnabled = true;
        }
  }
```

Código 3: Implementación del ejemplo anterior.

A continuación se muestra el resultado de hacer clic en el botón.

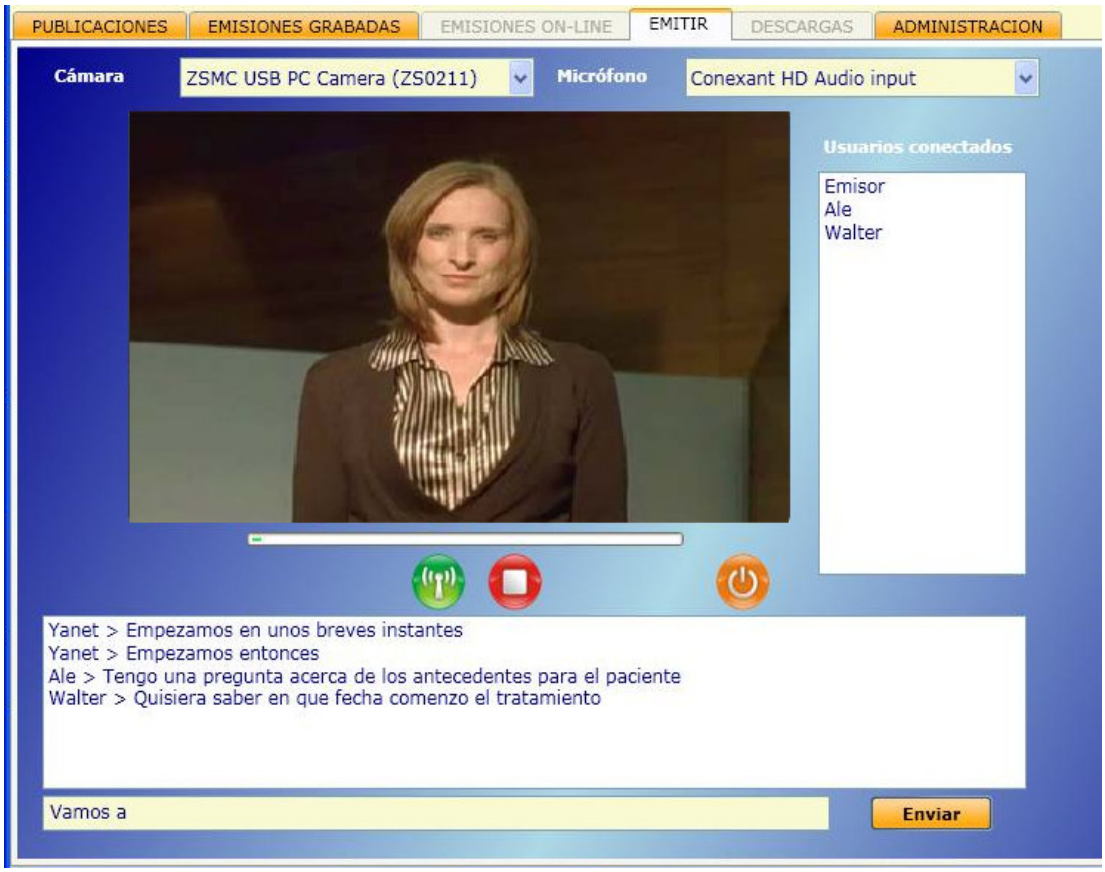


Figura 4: Resultado del ejemplo.

2.3 Tecnología común para interfaces de usuario de Windows y explorador web

Es importante crear interfaces de usuario eficaces para las aplicaciones de Windows y para las aplicaciones basadas en Web.

Hasta ahora, la generación de este tipo de interfaz requería el uso de un conjunto de tecnologías totalmente distintas a las usadas en la creación de interfaces nativas de Windows.

Por ello, los desarrolladores suelen centrarse en uno de los siguientes enfoques: si se trata de un desarrollador de interfaces de Windows o de un desarrollador de interfaces web. Por lo que se deduce que no existe motivo inherente que impida el uso de las mismas tecnologías tanto para interfaces nativas de Windows como para interfaces de explorador web.

WPF ofrece esta posibilidad. Así, un desarrollador puede crear una aplicación XAML del explorador (XBAP) con WPF, que se ejecuta en Internet Explorer. De hecho, es posible usar el mismo código para crear una aplicación de WPF independiente y una XBAP.

Cabe destacar que para el desarrollo de *G-Sell* solo se desarrollaron interfaces de usuarios eficaces para aplicaciones de Windows.

3. Diseño y controles

WPF ofrece un completo conjunto de características de programación de aplicaciones, algunas de ellas son:

- Lenguaje de marcado de aplicaciones extensibles (XAML).
- Controles.
- Enlace de datos.
- Diseño.
- Gráfico 2D y 3D.
- Animación, estilo, plantillas, documentos, multimedia, texto y tipografía.

Al crear una interfaz de usuario, se organizan los controles según su ubicación y tamaño para crear un diseño.

En el desarrollo de *G-Sell* se hizo uso de algunas de esas características, permitiendo así, lograr enriquecer sus interfaces.

El sistema de diseño administra la negociación entre los controles para determinar el diseño. Se expone además a los controles secundarios mediante las clases base de WPF. Para los diseños comunes, como son las cuadrículas, el apilamiento y el acoplamiento, WPF incluye varios controles de diseño:

- MediaElement: Es capaz de reproducir video en formato wmv, mpeg y avi junto con varios formatos de audio presenta la flexibilidad suficiente para constituir la base de un reproductor multimedia personalizado.
- StackPanel: los controles secundarios se apilan vertical u horizontalmente.
- DockPanel: los controles secundarios se alinean con los bordes del panel.
- Canvas: los controles secundarios proporcionan su propio diseño.
- Grid: los controles secundarios se sitúan por filas y columnas.
- StackPanel: los controles secundarios se organizan en una vista de una sola línea en sentido horizontal o vertical.
- WrapPanel: los controles secundarios se sitúan por orden de izquierda a derecha y se ajustan a la línea siguiente cuando hay más controles de los que caben en la línea actual.

A continuación se desplegarán ejemplos de algunos controles y estilos usados en la creación de *G-Sell*.

3.1 MediaElement

El marcado XAML siguiente implementa un reproductor multimedia mediante el uso de control MediaElement.

```
<MediaElement
        x:Name="WinMainOnLineMedia"
        Margin="66,39,189,246" Grid.Column="1"
        MediaFailed="mediaElement_MediaFailed"
        MediaEnded="mediaElement_MediaEnded"/>
```

Código 4: Ejemplo XAML de control MediaElement.

A continuación se despliega el control MediaElement en acción.



Figura 5: Ejecución de control MediaElement.

3.2 StackPanel

Como ya se describió este control trabaja como una pila, en este caso hacia abajo. A continuación se despliega un ejemplo del código XAML generado al desarrollar la vista previa de una entrevista.

```
<!-- Imagen preview de la entrevista.  -->
<Border BorderBrush="Orange" BorderThickness="4" CornerRadius="5"
      Margin="6" >

  <!-- StackPanel que contiene la informacion de cada entrevista.  -->
  <StackPanel Orientation="Vertical" VerticalAlignment="Center" >

    <!-- Descripcion de la entrevista.  -->
    <TextBlock   Width="auto" Height="auto" Text="{Binding
      InterviewDesc}" Style="{StaticResource TextBlockView}" />

    <!-- Usuario que publico la entrevista.  -->
    <TextBlock Width="auto" Height="auto" Text="{Binding UserDesc}"
      Style="{StaticResource TextBlockView}" />

    <!-- Imagen preview de la entrevista.  -->
    <Image Source="{Binding Path=FileJpg}"  Stretch="Fill"  Width="120"
      Height="120"    />

    <!-- Descargas y ranking de la entrevista.  -->
    <TextBlock  Width="auto" Height="auto"
      Text="{Binding QtyDownloadAndRanking}"
      Style="{StaticResource TextBlockView}" />

  </StackPanel>
```

Código 5: Ejemplo XAML de control StackPanel.

Produciendo el siguiente resultado:



Figura 6: Ejecución del control StackPanel.

3.3 DockPanel

El siguiente código XAML es generado al dockear arriba etiqueta publicaciones, dockear abajo botón publicar, y el resto el uso de flowdocumentpageviewer.

```
<DockPanel  HorizontalAlignment="Stretch" Height="{Binding
                  ElementName=TabItemPublicaciones, Path=Height}">
      <Canvas  DockPanel.Dock="Top" Height="30">
            <Label Style="{StaticResource LabelStd}"  Width="180"
                  Canvas.Left="0" Canvas.Top="0">Publicaciones
            </Label>
      </Canvas>
      <Canvas  DockPanel.Dock="Bottom" Height="30">
            <Button  Name="btnPublication"   Click="btnPublication_Click"
                  Style="{StaticResourceButtonStd}"
                  Canvas.Left="0"Canvas.Top="0"IsEnabled="False">Publicar
            </Button>
      </Canvas>
      <FlowDocumentPageViewer Name="DPVPublications" Width="Auto"
            HorizontalAlignment="Stretch"
             Style="{StaticResource FlowDocumentPageViewerStd}" >
      </FlowDocumentPageViewer>
</DockPanel>
```

Código 6: Ejemplo XAML de control DockPanel.

A continuación se despliega el resultado del código XAML antes descripto.

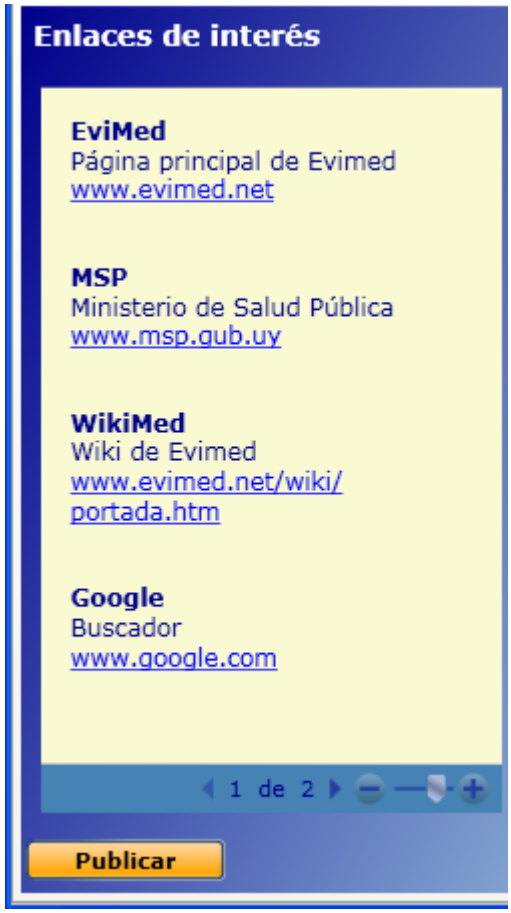


Figura 7: Ejecución del control DockPanel.

3.4 Canvas

Mediante este control se indica la posición de imágenes, botones, etiquetas, etc. El código XAML siguiente muestra un ejemplo de canvas utilizado en la aplicación.

```
<Canvas Name="CanvasDescargada"  Background="Transparent" Visibility="Hidden">
       <!--Titulos.-->
       <Label  Name="lblDescargadaPortada"  Style="{StaticResource LabelStd}"
              Canvas.Top="165" Canvas.Left="10">Portada de la Entrevista
       </Label>
       <Label  Name="lblDescargadaCierre"  Style="{StaticResource LabelStd}"
              Canvas.Top="165" Canvas.Left="10">Cierre de la Entrevista
       </Label>
       <!--Propiedades fijas de las entrevistas.-->
       <Image Name="ImageDescargada"   Width="150" Height="150" Canvas.Top="10"
              Canvas.Left="10"   />
       <Label Name="lblDescargadaInterviewDescL"
              Style="{StaticResource LabelEdit}"
              Canvas.Top="10" Canvas.Left="170">Descripción de la Entrevista:</Label>
       <Label  Name="lblDescargadaInterviewDesc"  Style="{StaticResource
              LabelEdit}"
              Canvas.Top="10" Canvas.Left="380"></Label>
       <Label  Name="lblDescargadaUserDescL"  Style="{StaticResource
              LabelEdit}"
              Canvas.Top="35" Canvas.Left="170">De:</Label>
       <Label  Name="lblDescargadaUserDesc"  Style="{StaticResource LabelEdit}"
              Canvas.Top="35" Canvas.Left="380"></Label>
       <Label  Name="lblDescargadaDateAndTimeL"
              Style="{StaticResource LabelEdit}" Canvas.Top="60"
              Canvas.Left="170">Fecha de la Entrevista:</Label>
       <Label  Name="lblDescargadaDateAndTime"  Style="{StaticResource
              LabelEdit}"
              Canvas.Top="60" Canvas.Left="380"></Label>
       <Label  Name="lblDescargadaFileSizeL"  Style="{StaticResource
              LabelEdit}"
              Canvas.Top="85" Canvas.Left="170">Tamaño:</Label>
       <Label  Name="lblDescargadaFileSize"  Style="{StaticResource LabelEdit}"
              Canvas.Top="85" Canvas.Left="380"></Label>
       <Label  Name="lblDescargadaFileLengthL"  Style="{StaticResource
              LabelEdit}"
              Canvas.Top="110" Canvas.Left="170">Duración:</Label>
       <Label  Name="lblDescargadaFileLength"  Style="{StaticResource
              LabelEdit}"
              Canvas.Top="110" Canvas.Left="380"></Label>
       <Label  Name="lblDescargadaGuestUsersDesc"  Style="{StaticResource
              LabelEdit}"
              Canvas.Top="135" Canvas.Left="380" ></Label>
       <!--Botones.-->
       <Button  Name="btnVerDescargada"  Canvas.Left="700"    Canvas.Top="110"
              Style="{StaticResource ButtonStd}"
              Click="btnVerDescargada_Click">Ver Emisión</Button>
       <Button  Name="btnCerrarDescargada"  Canvas.Left="700" Canvas.Top="135"
              Style="{StaticResource ButtonStd}"
              Click="btnCerrarDescargada_Click">Cerrar</Button>
</Canvas>
```

Código 7: Ejemplo XAML de control Canvas.

Su ejecución devuelve el siguiente resultado, como se despliega en la figura:

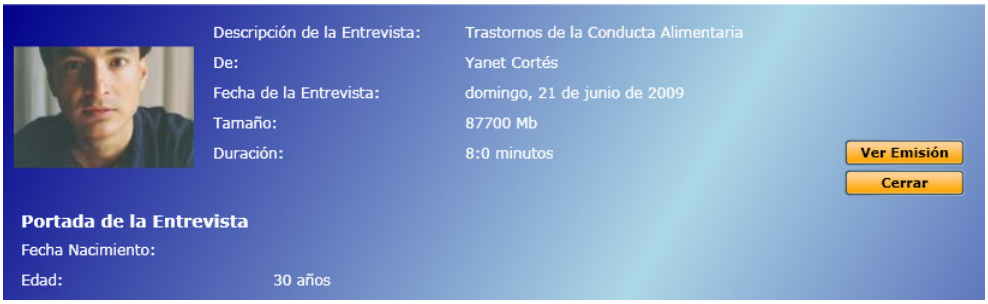


Figura 8: Ejecución del control Canvas.

3.5 Grid

En el código XAML desplegado a continuación se muestra el uso de este control en la aplicación desarrollada, definiendo botones, imágenes, lista de valores, etiquetas, etc.

```
<Grid Height="588" Width="693" Canvas.Top="0" Canvas.Left="0">
      <Grid.ColumnDefinitions>
             <ColumnDefinition Width="9*" />
             <ColumnDefinition Width="673*" />
             <ColumnDefinition Width="0*" />
             <ColumnDefinition Width="0*" />
             <ColumnDefinition Width="11*" />
      </Grid.ColumnDefinitions>
       <ListBox Margin="12,0,9,52" Name="WinMainOnLineListUserMsg"
            Height="138"
            VerticalAlignment="Bottom"  Grid.Column="1" />
        <ButtonName="WinMainOnLineEnviar" VerticalAlignment="Bottom"
                Click="WinMainOnLineEnviar_Click" Grid.Column="1"
                HorizontalAlignment="Right"
                Style="{StaticResourceButtonStd}"
                Margin="0,0,0,24">Enviar</Button>
        <TextBox Height="23" Margin="14,0,138,24"
                   Name="WinMainOnLinetxbMsg"
                   VerticalAlignment="Bottom"
                   Grid.Column="1" KeyDown="OnKeyDownUserHandler" />
        <Image Grid.Column="1" Margin="0,38,188,245"
               Name="WinMainOnLineImageBackground"
               Source="/Gesell.Client;component/imagenes/todo
               negro.png" Stretch="Fill" HorizontalAlignment="Right"
               Width="419" />
        <MediaElement x:Name="WinMainOnLineMedia"
                     Margin="67,39,189,246" Grid.Column="1"
                     MediaFailed="mediaElement_MediaFailed"
                     MediaEnded="mediaElement_MediaEnded"/>
       <ListViewMargin="0,54,11,211" Name="WinMainOnLineListUserLog"
                 Grid.Column="1" HorizontalAlignment="Right"
                 Width="144" />
</Grid>
```

Código 8: Ejemplo XAML de control Grid.

La figura siguiente muestra la ejecución del código XAML descripto.

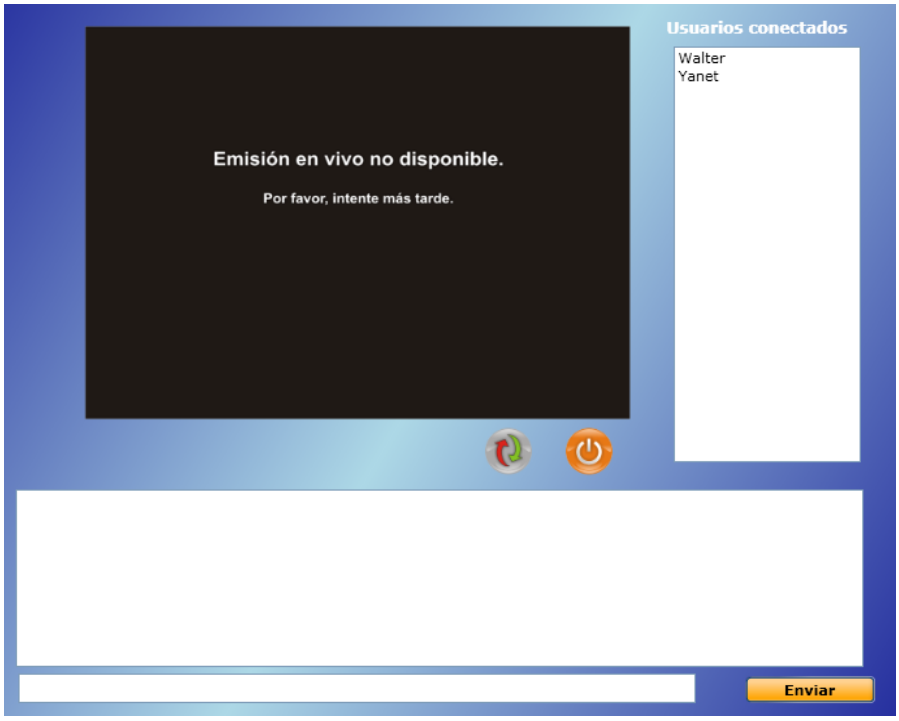


Figura 9: Ejecución del control Canvas.

3.6 FlowDocumentPageViewer

La información se puede cargar por xaml pero también puede cargarse por código. En el siguiente código se muestra como las publicaciones se cargan de acuerdo a los datos que obtiene de los objetos.

```
//Crea el documento.
 FlowDocument objFlowDocument = new FlowDocument();
 objFlowDocument.Background =
              System.Windows.Media.Brushes.LightGoldenrodYellow;
 objFlowDocument.FontFamily = new
              System.Windows.Media.FontFamily("Verdana");
 objFlowDocument.FontSize = 12;
 foreach (WebPublications objPublication in listWebPublications)
 {
 //Crea el párrafo.
       Paragraph objParagraph = new Paragraph();
       objParagraph.Inlines.Add(new Bold(new
              Run(objPublication.Title + "\n")));
       objParagraph.Inlines.Add(new
              Run(objPublication.Comment + "\n"));
       objParagraph.Inlines.Add(new
              Run(objPublication.Authors + "\n"));
        objParagraph.Inlines.Add(new
              Run(objPublication.DateInserted + "\n"));
        Hyperlink HyperlinkTemp = new Hyperlink(new
              Run(objPublication.Link + "\n"));
        HyperlinkTemp.NavigateUri = new
              System.Uri(objPublication.Source +
         objPublication.Link);
              HyperlinkTemp.Click += new
         RoutedEventHandler(Hyperlink_Click);
              objParagraph.Inlines.Add(HyperlinkTemp);
       //Agrega el párrafo al documento.
       objFlowDocument.Blocks.Add(objParagraph);
  }
  // Asocia el documento al control.
  DPVPublications.Document = objFlowDocument;
```

Código 9: Ejemplo XAML de control FlowDocumentPageViewer.

La figura siguiente muestra su ejecución:

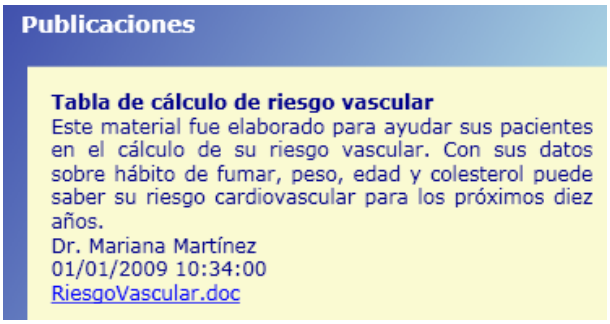


Figura 10: Ejecución del control FlowDocumentPageViewer.

3.7 Tab Control

Dicho control es utilizado para control de pestañas. A continuación se despliega el código XAML para el desarrollo de una de las pestañas implementadas en la aplicación.

```
<!-- Tabcontrol con pestañas para opciones del
programa. -->
<TabControl Height="Auto"  Name="tabMain"
      HorizontalContentAlignment="Stretch"
      VerticalContentAlignment="Stretch"  >

<!--Pestañas de publicaciones. -->
     <TabItem Name="TabItemPublicaciones"
      Header="PUBLICACIONES"
      Style="{StaticResource TabItemStd}" >
```

Código 10: Ejemplo XAML del control Tab.

La ejecución del código anterior se ve reflejada en la siguiente figura:

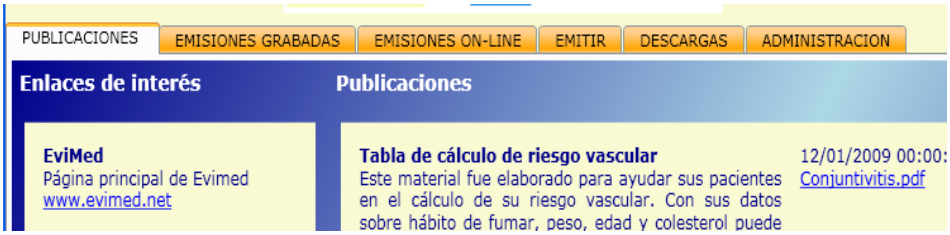


Figura 11: Ejecución del control Tab.

3.8 Wrap panel

Es asociado a un listbox para mostrar las imágenes previas. Cuando la ventana cambia tamaño las imágenes se reacomodan solas. El código desplegado a continuación es un ejemplo de ello.

```
<Setter Property="ItemsPanel">
 <Setter.Value>
  <ItemsPanelTemplate>
   <WrapPanel ScrollViewer.VerticalScrollBarVisibility="Visible"/>
  </ItemsPanelTemplate>
 </Setter.Value>
</Setter>
<Setter  Property="ScrollViewer.HorizontalScrollBarVisibility"
Value="Disabled"  />
</Style>
</StackPanel.Resources>

<!—Contenedor de las entrevistas. -->
<ListBox Name="ListBoxGrabadas" ItemsSource="{Binding}"
   MouseDoubleClick="MouseDoubleClick_ListBoxGrabadas"
   Background="Transparent"  BorderBrush="Transparent"     />
</StackPanel>
```

Código 11: Ejemplo XAML del control Wrap panel.

Su ejecución se ve así:



Figura 12: Ejecución del control Wrap panel.

3.9 Binding de controles con datos

A continuación se despliega el código que permite bindear una grilla con los datos de una lista de objetos.

```
grdUsers.DataContext = listUsers;

grdUsers.Columns.ElementAt(COL_USERID).Header = "Usuario";
grdUsers.Columns.ElementAt(COL_USERNAME).Header = "Nombre";
grdUsers.Columns.ElementAt(COL_USERSECONDNAME).Header = "Apellido";
grdUsers.Columns.ElementAt(COL_EMAIL).Header = "Correo Electrónico";
grdUsers.Columns.ElementAt(COL_PROFILE).Header = "Perfil";

grdUsers.Columns.ElementAt(COL_USERID).DisplayMemberBinding = new
      Binding("UserId");
grdUsers.Columns.ElementAt(COL_USERNAME).DisplayMemberBinding = new
      Binding("UserName");
grdUsers.Columns.ElementAt(COL_USERSECONDNAME).DisplayMemberBinding
      = new Binding("UserSecondName");
grdUsers.Columns.ElementAt(COL_EMAIL).DisplayMemberBinding = new
      Binding("Email");
grdUsers.Columns.ElementAt(COL_PROFILE).DisplayMemberBinding = new
      Binding("ProfileDescription");
```

Código 12: Binding de control con datos.

3.10 Crear estilos para controles

WPF permite especificar el aspecto de los controles de manera global, en lugar de establecer las propiedades en cada instancia en la aplicación, creando un Style. Algunos estilos definidos en G-Sell son:

- Estilos para definir botones: de color naranjo, color con degradé, ancho de botón, fuente, botones aceptar y cancelar.
- Estilos para definir textbox: tipo de letra, estilo, tamaño de control y de letra, color de letra y de fondo, etc.

- Etiquetas: letra blanca, transparente para que se el fondo sea visto.
- Se describen a continuación los códigos XAML que establecen algunos de estos estilos:

```
<!--Estilos para textbox editables.-->
<Style x:Key="TextBoxEdit" TargetType="TextBox">
    <Setter Property="FontFamily" Value="Verdana"/>
    <Setter Property="FontSize" Value="12px"/>
    <Setter Property="Foreground" Value="DarkBlue"/>
    <Setter Property="Background" Value="LightGoldenrodYellow"/>
    <Setter Property="Height" Value="25"/>
    <Setter Property="Margin" Value="1"/>
    <Setter Property="FontWeight" Value="Normal"/>
    <Setter Property="Width" Value="300"/>
    <Setter Property="Height" Value="25"/>
</Style>

<!--Estilos para etiquetas de títulos.-->
<Style x:Key="LabelStd" TargetType="Label">
    <Setter Property="FontFamily" Value="Verdana"/>
    <Setter Property="FontSize" Value="14px"/>
    <Setter Property="Foreground" Value="White"/>
    <Setter Property="Background" Value="Transparent"/>
    <Setter Property="Height" Value="27"/>
    <Setter Property="Margin" Value="1"/>
    <Setter Property="FontWeight" Value="Bold"/>
</Style>

<!--Estilos para botones.-->
<Style x:Key="ButtonStd" TargetType="Button">
    <Setter Property="Margin" Value="2"/>
    <Setter Property="FontFamily" Value="Verdana"/>
    <Setter Property="FontSize" Value="11px"/>
    <Setter Property="FontWeight" Value="Bold"/>
    <Setter Property="Width" Value="100"/>
    <Setter Property="Background" >
        <Setter.Value>
            <LinearGradientBrush StartPoint="0,0" EndPoint="0,1" >
                <GradientStop Color="#FFFFD190" Offset="0.2"/>
                <GradientStop Color="Orange" Offset="0.85"/>
                <GradientStop Color="#FFFFD190" Offset="1"/>
            </LinearGradientBrush>
        </Setter.Value>
    </Setter>
</Style>
```

Código 13: Ejemplo XAML del control de estilos.

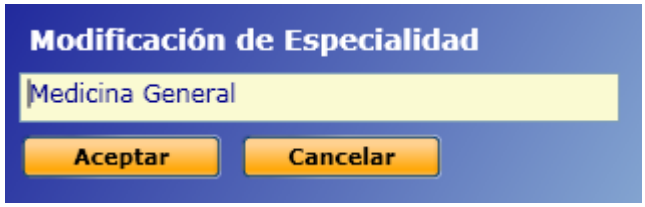


Figura 13: Ejecución de estilos.

Los estilos definidos arriba son asignados como se despliega en el siguiente código: Style='nombre'

```
<Canvas Name="CanvasSpecialities">
       <Label Name="lblSpecialityTitle"
              Style="{StaticResource LabelStd}" Canvas.Top="10"
              Canvas.Left="0">Alta de
       Especialidad</Label>
       <Label Canvas.Left="0" Canvas.Top="40"  Name="lblDescription"
              Style="{StaticResource LabelEdit}">Descripción</Label>
       <TextBox Canvas.Left="0" Canvas.Top="40" Name="txtDescription"
              MaxLength="40" Style="{StaticResource TextBoxEdit}"/>
       <Button Canvas.Left="110" Canvas.Top="70"
              Name="btnCancelSpeciality" Click="btnCancelSpeciality_Click"
              Style="{StaticResource ButtonStd}" >Cancelar</Button>
       <Button Canvas.Left="0" Canvas.Top="70"
              Name="btnAceptSpeciality" Click="btnAceptSpeciality_Click"
              Style="{StaticResource ButtonStd}" >Aceptar</Button>
</Canvas>
```

Código 14: Ejemplo XAML de asignación de estilos.

4. Referencias

- Control FlowDocumentPageViewer – último acceso: 29/06/09
  http://msdn.microsoft.com/es-es/library/ms742548.aspx
- Ejemplos y controles WPF – último acceso: 29/06/09
  http://msdn.microsoft.com/es-es/library/aa663364.aspx
- Generalidades de WPF – último acceso: 29/06/09
  http://es.wikipedia.org/wiki/Windows_Presentation_Foundation
- Windows Presentation Foundation – último acceso: 29/06/09
  http://msdn.microsoft.com/es-es/library/ms754130.aspx

4. Introducción a Windows Communication Foundation (WCF)

1. Principios de diseño

1- Unificar tecnologías existentes.

2- Aumentar la productividad.

3- Permitir la integración con otras tecnologías de mensajería ya sea de otros fabricantes o de la propia MS.

2. Concepto

Se trata de un cliente y un servicio que intercambian mensajes a través de un endpoint (puntos de acceso).

Si el cliente puede crear un endpoint compatible con alguno de los endpoints expuestos por el servidor. Entonces la comunicación es posible.

Los servicios pueden estar hospedados en un Internet Information Services (IIS) pero también en una aplicación por ejemplo, una aplicación WinForms.

3. Abc de WCF

El ABC en WCF es una regla nemotécnica que permite comprender de manera fácil tres conceptos importantísimos en WCF. Endpoint (punto de acceso) =Una dirección (Address), enlace (Bindings) y un contrato (Contract).

3.1 Address

Indica donde está el servicio, se traduce en una URI del tipo de los servicios Web, estas URIs puede direccionar los siguientes protocolos:

1- HTTP

2- TCP

3- NamedPipe

4- Peer2Peer

5- Msmq

3.2 Bindings

Un binding específica cómo se accede al servicio, esto es mediante tres partes:

1- Un protocolo de transporte (por ejemplo HTTP, TCP).

2- Una codificación determinada (por ejemplo binario o soap).

3- Uno o más protocolos de funcionamiento (por ejemplo transacciones).

3.3 Contract

Un contrato indica qué ofrece un servicio, es decir que métodos y operaciones expone.

En Windows Communication Foundation (WCF) hay tres tipos de contratos:

1- Service Contract: Define los métodos que un servicio expone a sus clientes.

2- Data Contract: Define la composición de los parámetros de los métodos definidos en el service contract.

3- Message Contract: Es el tipo de contrato que no suele usarse. Permite especificar a nivel de mensaje que datos van en la cabecera y en el cuerpo.

Definición de contratos

- Para Service Contracts: definir una interfaz y aplicar el atributo [OperationContract] a cada método que se desea que el servicio exponga, sólo los métodos que tengan ese atributo serán expuestos por el servicio a sus clientes.
- Para Data Contract: definir una clase y aplicar el atributo [DataMember] a cada miembro de la clase que deba ser serializado, y por lo tanto pasados/recibidos a/desde el servicio cuando se opere con datos de esa clase.

4. Servicios

Para un servicio pueden existir diferentes combinaciones de ABC, por ejemplo un servicio puede exponer n operaciones (métodos) y pueden existir muchas combinaciones ABC para un cierto servicio.

Por ejemplo exponer operaciones con un protocolo distinto y mediante un URL diferente, cada combinación se denomina endpoint, por ende un servicio WCF puede tener múltiples endpoints.

La figura a continuación ilustra los conceptos antes mencionados.

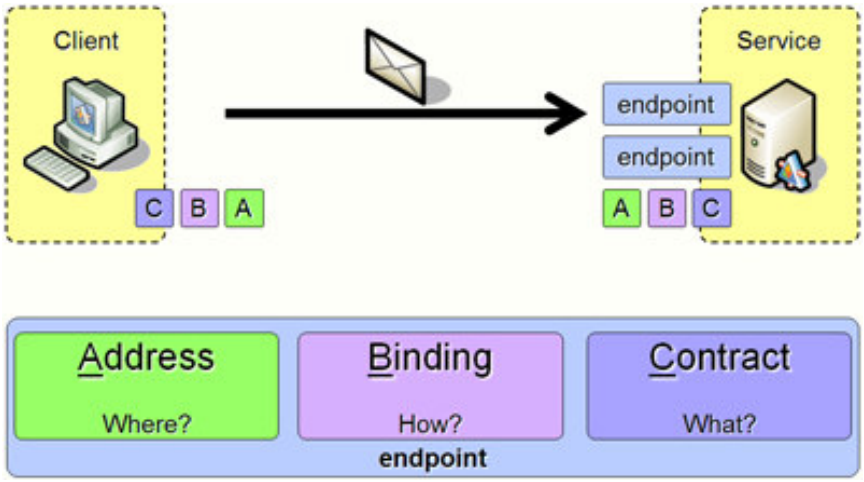


Figura 1: Conceptos WCF.

5. Behaviours

Los behaviours no forman parte de los endpoints, son elementos locales al cliente o al servidor.

Permiten controlar aspectos relativos a la infraestructura de WCF:

1- Cómo el servicio expone su metadata (wsdl).

2- Gestión de las instancias del servicio (singleton, un servicio por cliente, un servicio por mensaje).

3- Máximo número de clientes admitidos.

4- Información de enrutamiento.

6. Referencias

- Conceptos en WCF – último acceso 28/06/09
  http://geeks.ms/blogs/oalvarez/archive/2006/11/27/comienzo-con-wcf.aspx
- Introducción WCF – último acceso 28/06/09
  http://techedbyraona.blogspot.com/2006/11/dev214-introducing-wcf.html

5. Publicación y descarga de archivos

1. Introducción

Uno de los puntos importantes a resolver al implementar la solución virtual para *G-Sell* es la transferencia de archivos como forma de compartir la información generada por los distintos usuarios. Así, el resultado de las entrevistas generadas en los clientes, como los archivos de interés, son subidos a un sitio que centraliza y procesa cada uno poniéndolo luego a disposición de la comunidad. Este comportamiento cliente/servidor, donde un cliente publica cierta información en un servidor que luego queda disponible para ser descargada es el que se quiere modelar en la publicación y descarga de archivos. Basado en un ejemplo publicado en CodeProject (2007) se decidió implementar un servicio que permita soportar la funcionalidad descrita.

Como forma de aprovechar las facilidades que brindan los servicios Web para atravesar firewalls y considerando el ambiente real donde se va a desempeñar la solución planteada se eligió construir un servicio Web que oficie como servidor de archivos.

Dicho servicio está implementado en WCF y es alojado en un servidor de aplicaciones Internet Information Service IIS versión 6.0 ubicado en el sitio que se va a usar como servidor.

En la aplicación cliente se programan las funciones de publicación y descarga en hilos independientes entre si e independientes también al que ejecuta la interfaz de usuario. Esto permite adoptar cierta simultaneidad de tareas que permite descargar una entrevista viendo su progreso, mientras realizan otras operaciones.

Los tres archivos (audio/video, chat y portada) que componen una entrevista deben ser publicados en el servidor. Para ello, son empaquetados en un archivo con formato y extensión zip encriptado con contraseña utilizando la librería DotNetZip. Esta librería es utilizada por la aplicación cliente y por el servicio Web para dar soporte al empaquetado y desempaquetado de archivos.

A modo de mejorar la experiencia del usuario al publicar una entrevista y sacando partido al hecho que una importante cantidad de información en la entrevista se trata de video, el archivo con formato zip generado es realizado sin el proceso de compresión de datos.

2. Cliente

La aplicación cliente es la que consume el servicio, por lo que en las referencias a servicios se agrega una que es llamada FileTransferClient. Esta referencia es el proxy del servicio Web en el extremo cliente que le permite a este enviar pedidos al servidor ocultando la complejidad del servicio Web y los protocolos SOAP.



Figura 1: Referencia FileTransferClient.

Al agregar la referencia automáticamente se genera la configuración del servicio en el extremo cliente. Como se trata de un servicio Web para transferencia de archivos es necesario modificar las propiedades de timeout para soportar mayores demoras.

```
<binding name="BasicHttpBinding_IFileTransferService"
       closeTimeout="00:01:00" openTimeout="00:01:00"
       receiveTimeout="00:10:00" sendTimeout="00:05:00"
       maxReceivedMessageSize="10067108864"
       messageEncoding="Mtom"
   transferMode="Streamed" useDefaultWebProxy="true" >
```

Código 1: Configuración de FileTransferService en app.config.

La publicación y descarga de archivos implementada en la aplicación se puede clasificar según el tipo de archivo a ser subido. En el proceso de publicación de entrevistas la aplicación cliente, en forma transparente para el usuario, se encarga de empaquetar los tres archivos generados en uno que luego envía al servidor de archivos.

La publicación de artículos o documentos de interés, en cambio, se hace directamente mediante el uso del método UploadFile del servicio Web.

En el caso de subida y descarga de entrevistas se necesitan realizar tareas que permitan procesar la información, por lo que en este caso se definen métodos que usan UploadFile y DownloadFile como base.

Como ya se mencionó, tanto el empaquetado de archivos seguido del envío, la descarga seguida del desempaquetado, como las publicaciones y descargas de archivos comunes se implementan en hilos independientes al que ejecuta la interfaz de usuario de la aplicación.

En toda solución de estas características resulta importante brindar al usuario final una noción del progreso que está teniendo la acción que ha decidido ejecutar. Para esto se necesita conocer la cantidad de Bytes compactados, subidos o descargados.

En la descarga de archivos la aplicación cliente tiene total control del procedimiento y es implementada usando un loop que lee el flujo de datos desde el servidor Byte por Byte. De esta forma, el cliente conoce que cantidad de flujo ha sido leído, así como cuánto resta por descargar y puede representarlo en forma gráfica fácilmente.

Cuando se publica el loop implementado reside en el servidor, por lo que para que el cliente pueda saber cuántos Bytes fueron leídos por el servidor, la aplicación cliente utiliza la clase StreamWithProgress que hereda de System.IO.Stream.

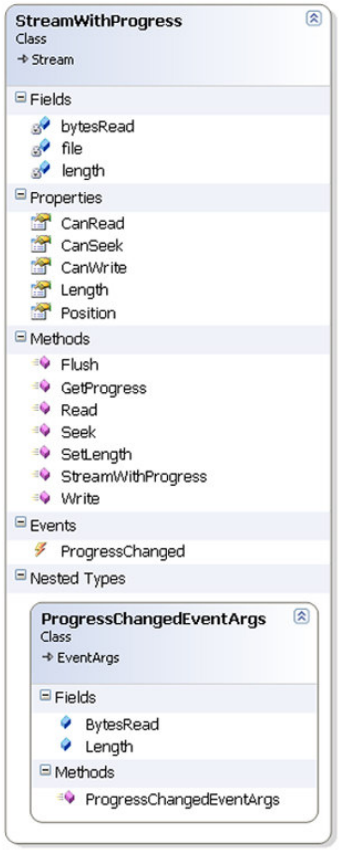


Figura 2: Diagrama de clase StreamWithProgress.

Una instancia de esta clase es pasada al servidor en lugar del stream original. Esta clase sobrescribe el método Read implementado en Stream reportando a través del evento ProgressChanged el progreso de la publicación.

```
public override int Read(byte[] buffer, int offset, int count)
{
int result = file.Read(buffer, offset, count);
bytesRead += result;
      if (ProgressChanged != null)
      ProgressChanged(this,
            new ProgressChangedEventArgs(bytesRead, length));
   return result;
}
```

Código 2: Reimplementación de método Read en StreamWithProgress.

2.1 Transferencia de entrevistas

Los archivos que componen una entrevista deben ser empaquetados para ser publicados en el servidor y desempaquetados una vez que son descargados. Para implementar este proceso se utiliza la librería DotNetZip y las operaciones ofrecidas por el servicio Web.

Publicación

En el momento que el cliente decide publicar una entrevista en el servidor, la aplicación automáticamente genera el archivo zip correspondiente. Por medio de los métodos ofrecidos en la librería DotNetZip se configuran la clave, el algoritmo de encriptación, el tipo de compresión y demás parámetros de dicho archivo formato zip.

A continuación la información de la entrevista empaquetada, esto es el archivo zip recién creado, es subida por medio del servicio Web a la carpeta destino definida en el sistema.

Una vez concluida esta acción y haciendo uso de la operación ExtractJPG del mismo servicio, se extrae del paquete la imagen de portada de la entrevista.

Como resultado de este proceso queda disponible en el servidor la información necesaria de la entrevista para que luego sea descargada a demanda por los demás usuarios.

El método para implementar la publicación de la entrevista invoca UpLoadZipFile que se encarga de realizar todo el trabajo antes descrito, valiéndose de las operaciones que ofrece el servicio Web.

```
private void UpLoadZipFile(string baseFileName)
{
      // Compacto emision a enviar al servidor.
   EmpaquetarEmision(baseFileName);
      …
      // Subo el archivo recibido
      client.UploadFile(fileInfo.Name, fileInfo.Length, pathDestino,
                  uploadStreamWithProgress);

      // Luego de subido en el servidor, obtengo imagen de portada.
      client.ExtractJPG(baseFileName);

   // Cierro el servicio cliente
   client.Close();
      …
```

Código 3: Sección de método UpLoadZipFile.

En el método auxiliar EmpaquetarEmision se procesa el contenido de la entrevista definiendo los parámetros correspondientes al archivo con formato zip.

```
…
using (var zip = new ZipFile())
{
    zip.Password = "Camara.Gesell!";
    zip.Encryption = EncryptionAlgorithm.PkzipWeak;
    zip.UseZip64WhenSaving = Zip64Option.AsNecessary;
    zip.CompressionLevel = Ionic.Zlib.CompressionLevel.NONE;
                …
    zip.Save(emision +".zip");
…
```

Código 4: Sección de método EmpaquetarEmision.

La ejecución de todo el proceso de publicación de la entrevista puede no ser inmediata, por lo que es necesario definir un nuevo hilo y darle al usuario una idea sobre el progreso de cada una de las tareas. Esto último es logrado actualizando constantemente una barra de progreso indicando su evolución. Para realizarlo se programan eventos que son los encargados de calcular y actualizar los valores presentados al usuario final.

Debido a que los eventos de actualización y el control de usuario que marca el progreso están en distintos hilos es necesario trabajar con el Dispatcher.BeginInvoke para que dicha barra de progreso refleje los avances.

Descarga

El proceso de descarga de entrevistas publicadas en el servidor es controlado completamente por la aplicación cliente. Esta invoca al método DownLoadFile iniciando una transferencia a través del servicio Web que descarga el archivo zip desde el servidor y lo almacena temporalmente en la carpeta predeterminada.

```
…
  // Baja archivo desde el servidor
  DownLoadFile(fileName, origen);

  // Descomprime el archivo descargado
  OpenFile(fileName);
       …
```

Código 5: Sección del evento DownloadButton_Click.

Una vez descargada la entrevista, se llama el método OpenFile para desempaquetarla, obteniendo, de esta forma, los tres archivos correspondientes. Luego de abierto el archivo zip, este es eliminado con el fin de mantener un uso adecuado del espacio en disco.

Como ya se adelantó la descarga es controlada por el cliente a través de un loop donde se va leyendo el archivo byte por byte. El método DownLoadFile es el encargado de esta tarea y de ir actualizando la barra de progreso a medida que esta transcurre.

```
private void DownLoadFile(string fileName, string origen)
{

        …
do
{
        // Leo bytes desde el stream de entrada
        int bytesRead = inputStream.Read(buffer, 0, chunkSize);
        if (bytesRead == 0) break;

        // Escribo bytes en el stream de salida
        writeStream.Write(buffer, 0, bytesRead);

        position = writeStream.Position;

        (actualiza progressBar con position)

} while (true);

writeStream.Close();
…
```

Código 6: Sección del método DownLoadFile.

En la siguiente sección de código se ilustra el proceso de extracción de la información usando DotNetZip y la forma en la que es asociado el evento zip_ExtractProgress para notificar al usuario el avance del proceso de desempaquetado.

```
private void OpenFile(string zipFile)
{
                …
  using (ZipFile zip = ZipFile.Read(filePath))
  {
        zip.ExtractProgress += zip_ExtractProgress;
        foreach (ZipEntry e in zip)
        {
            e.ExtractWithPassword(path, true, "Camara.Gesell!");
        }
```

Código 7: Sección del método OpenFile.

Origen y destino de archivos

Las entrevistas grabadas, compuestas por el chat, video y portada, son almacenadas en el directorio Emisiones ubicado en la carpeta de instalación de la aplicación. Este directorio es el que se toma como origen de los datos en el proceso de publicación de entrevistas.

En la carpeta de instalación de la aplicación también se cuenta con un directorio Download donde se almacenan todas las descargas realizadas por el usuario. Cuando las entrevistas son descargadas con el objetivo de ser vistas a demanda tienen como destino una subcarpeta de Download llamada Emisiones.

2.2 Transferencia de archivos comunes

La aplicación también cuenta con la posibilidad de publicar artículos o documentos de interés. Estos artículos y cualquier otro archivo común que necesite ser subido al servidor son transferidos por medio de la operación UploadFile que ofrece el servicio Web.

3. Servidor

El modelo de servicio elegido en este caso para representar el servidor de archivos es basicHTTPBinding.

El servicio Web WCF implementado y alojado en el servidor Internet Information Services es llamado *FileTransferService.* Su definición puede ser dividida en tres secciones, configurando la forma que el servicio se comporta y cómo es expuesto al resto del mundo.

3.1 Configuración

La configuración del servicio se define en el archivo Web.config que es ubicado en la raíz del directorio asociado a dicho servicio en el servidor IIS.

Lo más destacable en la configuración del servicio es que este usa MTOM para codificar los mensajes y Stream como modo de transferencia.

La propiedad maxReceivedMessageSize define el máximo tamaño de mensaje transferido por el servicio, como se trata de implementar la transferencia de archivos necesitamos que el valor de esta propiedad sea un valor adecuado y grande.

```
<basicHttpBinding>
    <binding name="FileTransferServicesBinding"
             transferMode="Streamed"
             messageEncoding="Mtom"
             maxReceivedMessageSize="10067108864" >
    </binding>
</basicHttpBinding>
```

Figura 3: Sección de configuración en Web.config.

3.2 Contrato

El servicio que modela el servidor de transferencia incluye un archivo de código llamado *FileTransferServiceContract.cs* que contiene la definición completa de la interfaz que describe las operaciones provistas por el servidor.

```
[ServiceContract]
public interface IFileTransferService
{
        [OperationContract]
        void UploadFile(RemoteFileInfo request);

        [OperationContract]
        RemoteFileInfo DownloadFile(DownloadRequest request);

        [OperationContract]
        void ExtractJPG(string request);
}
…
```

Código 8: Vista de implementación del contrato.

Como se puede ver en el código anterior, se definen tres operaciones que el servicio de transferencia deberá brindar a los clientes.

DownloadFile

Este método recibe una instancia de DownloadRequest que contiene el nombre del archivo a ser descargado por el cliente. Como resultado devuelve una instancia de RemoteFileInfo (definida en este mismo archivo de código) que contiene el nombre y el tamaño en Bytes del archivo a ser descargado además del flujo de datos propiamente dicho. Esta instancia es la que finalmente usa el cliente para poder descargar el archivo.

UploadFile

El método UploadFile recibe una instancia del mensaje *RemoteFileInfo* similar a la usada en *DownloadFile*, salvo que en este caso la propiedad *length* no es necesaria.

ExtractJPG

Esta operación es implementada para permitir la extracción de la imagen de portada desde la entrevista empaquetada una vez que esta se encuentra en el servidor. Recibe como parámetro un string que representa el nombre de la entrevista a procesar.

3.3 Implementación

El servidor de archivos también incluye un archivo de código FileTransferService.svc.cs que contiene la implementación del contrato, por ende, el que realiza todo el trabajo en el extremo servidor. Aquí es donde se implementa la interfaz IFileTransferService que constituye en el contrato del servicio.

```
public class FileTransferService : IFileTransferService
{
                …
        }
```

Figura 4: Definición de FileTransferService.

La descarga de un archivo comienza después que la ejecución del método DownloadFile es completada, esto es una vez que el cliente obtiene la instancia RemoteFileInfo retornada por este método.

```
public RemoteFileInfo DownloadFile(DownloadRequest request)
{
        …
        //Abro el stream para la transferencia
        FileStream stream = new FileStream(filePath, FileMode.Open,
                                           FileAccess.Read);
        //Retorna el resultado
        RemoteFileInfo result = new RemoteFileInfo();
        …
        result.FileByteStream = stream;
        return result;
}
```

Código 9: Implementación de DownloadFile.

Luego que el cliente complete el proceso de descarga, el servidor debe cerrar el stream abierto. Para realizar esta tarea, se impone que RemoteFileInfo implemente la interfaz IDisposable y el stream es cerrado en el método Dispose correspondiente.

```
[MessageContract]
public class RemoteFileInfo : IDisposable
{
        …
public void Dispose()
{
        if (FileByteStream != null)
        {
                FileByteStream.Close();
                FileByteStream = null;
         }
}
```

Código 10: Implementación de Dispose en RemoteFileInfo.

En la publicación de archivos se dispone de un loop en el servidor que lee el flujo de datos y notifica al cliente sobre el progreso del mismo por medio de la reimplementación del método Read de la clase Stream.

La siguiente sección de código muestra los aspectos relevantes de la implementación de esta operación.

```
public void UploadFile(RemoteFileInfo request)
{
  …
      do
      {
            /* Lee bytes desde el stream de entrada
            * Por medio de override del metodo Read
            * actauliza barra de progreso en el cliente. */
            int bytesRead = request.FileByteStream.Read(buffer, 0,
                                                        chunkSize);
            if (bytesRead == 0) break;
            // Escribe bytes al stream de salida
            writeStream.Write(buffer, 0, bytesRead);
      } while (true);
      writeStream.Close();
      …
```

Código 11: Sección de método UpLoadFile.

El servicio Web necesita brindar soporte al desempaquetado de archivos para poder brindar al cliente la posibilidad de extraer la imagen de portada desde el paquete publicado. Para esto se incorpora al servicio la librería DotNetZip empaquetada en Ionic.Zip.dll que permite la manipulación de archivos con formato zip.

```
public void ExtractJPG(string request)
{
        …
        using (ZipFile zip = ZipFile.Read(filePath))
        {
                ZipEntry e = zip[fileToExtract]; /* overwrite = true */
                e.ExtractWithPassword(pathAbsoluto, true, "Camara.Gesell!");
        }
    …
```

Código 12: Implementación de la extracción de JPG.

3.4 Destino de archivos

El servidor de archivos debe tener un repositorio donde almacenar toda la información publicada por los clientes que luego de procesada será ofrecida para su descarga. La dirección absoluta de este repositorio es definida en la base de datos e inicialmente se configura relativa al directorio asociado al servicio Web.

En la imagen se puede apreciar la estructura básica de composición del servicio Web que implementa la publicación y descarga de archivos, mostrando su organización en las distintas carpetas definidas para el prototipo.

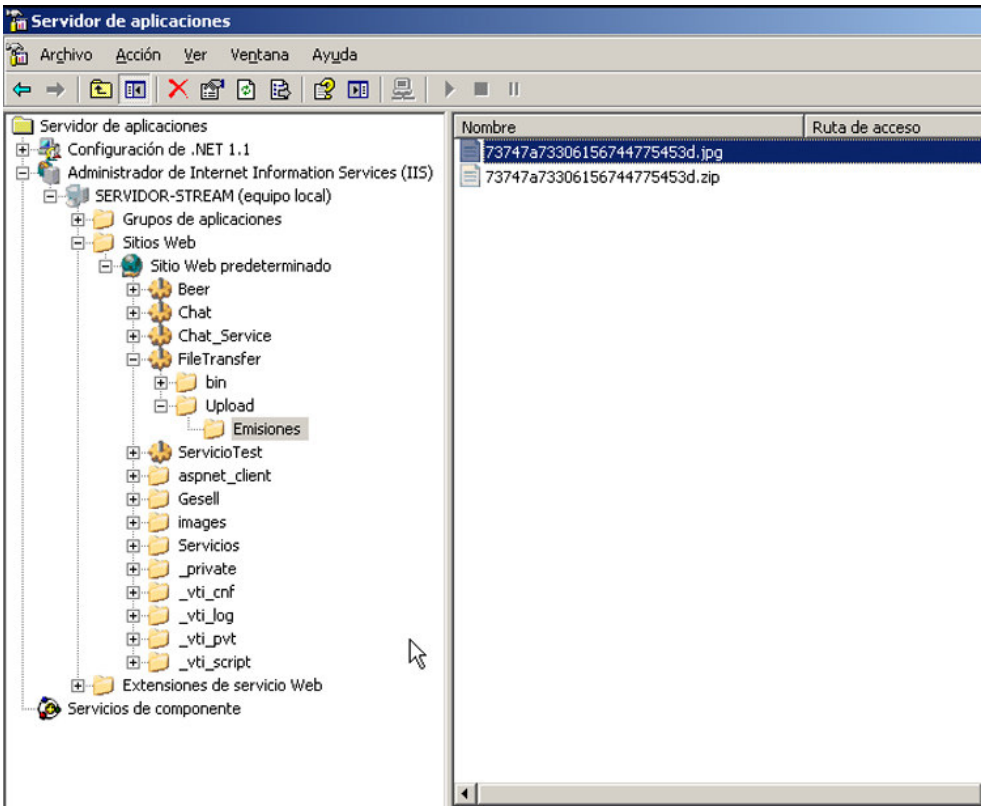


Figura 5: Estructura de servicio Web FileTransfer.

4. Librería DotNetZip

El producto DotNetZip es una librería de clases de fácil uso que permite a las aplicaciones la manipulación de archivos con formato zip. De esta forma, cualquier aplicación que sea desarrollada en .NET puede crear, leer, extraer y modificar archivos zip.

A continuación se detallan las operaciones que esta librería proporciona:

- Crear un paquete zip, agregando archivos o directorios en el paquete .
- Extraer archivos desde un paquete existente.
- Modificar un paquete existente, eliminando o agregando nuevas entradas.
- Proteger mediante clave entradas del paquete.
- Obtener entradas desde un archivo o un stream.
- Leer un archivo zip desde una archivo o un stream.
- Extraer entradas en un archivo o un stream.

La librería puede ser usada por cualquier aplicación .NET como Console, WinForms, WPF, ASP.NET, Sharepoint, servicios Web WCF, entre otras. Dicha librería produce archivos con formato zip que son completamente interoperables y compatibles con Windows Explorer, aplicaciones Java o incluso aplicaciones en entornos Linux. Los zips producidos pueden ser leídos y manejados por WinZip, WinRar y otras herramientas.

DotNetZip está empaquetada en una única DLL que pesa poco más de 300KB y no incluye dependencias de terceras partes. De esta forma para obtener los beneficios de compresión, empaquetado y descompresión de archivos solo es necesario referenciar a una DLL.

Por último, la librería brinda, entre otras características, soporte a zips con clave, Unicode, ZIP64, streams de entrada y salida, encriptación AES, múltiples niveles de compresión y paquetes autoextraíbles.

4.1 El namespace Ionic.Zip

Type	Description
BadCrcException	Issued when an CRC check fails upon extracting an entry from a zip archive.
BadPasswordException	Issued when an ZipEntry.ExtractWithPassword() method is invoked with an incorrect password.
BadReadException	Indicates that a read was attempted on a stream, and bad or incomplete data was received.
BadStateException	Indicates that an operation was attempted on a ZipFile which was not possible given the state of the instance. For example, if you call Save() on a ZipFile which has no filename set, you can get this exception.
CRC32	Calculates a 32bit Cyclic Redundancy Checksum (CRC) using the same polynomial used by Zip. This type is used internally by DotNetZip; it is generally not used directly by applications wishing to create, read, or manipulate zip archive files.
CrcCalculatorStream	A Stream that calculates a CRC32 (a checksum) on all bytes read, or on all bytes written.
EncryptionAlgorithm	An enum that provides the various encryption algorithms supported by this library.
ExtractProgressEventArgs	Provides information about the progress of the extract operation.
ReadProgressEventArgs	Provides information about the progress of a Read operation.

ReReadApprovalCallback	Delegate for the callback by which the application gives approval for multiple reads of the file stream. This callback is called only when the initial compression operation inflates the size of the file data.
SaveProgressEventArgs	Provides information about the progress of a save operation.
SelfExtractorFlavor	An enum that provides the different self-extractor flavors
SfxGenerationException	Issued when errors occur saving a self-extracting archive.
WantCompressionCallback	Delegate for the callback by which the application tells the libraary whether to use compression on the file or not.
Zip64Option	Options for using ZIP64 extensions when saving zip archives.
ZipEntry	This class models an entry in the directory contained within the zip file. The class is generally not used from within application code, though it is used by the ZipFile class. Represents a single entry in a ZipFile. Typically, applications get a ZipEntry by enumerating the entries within a ZipFile, or by adding an entry to a ZipFile.
ZipException	Base class for all exceptions defined by and throw by the Zip library.
ZipFile	The ZipFile type represents a zip archive file. This is the main type in the DotNetZip class library. This class reads and writes zip files, as defined in the format for zip described by PKWare. The compression for this implementation was, at one time, based on the System.IO.Compression.DeflateStream base class in the .NET Framework base class library, available in v2.0 and later of the .NET Framework. As of v1.7 of DotNetZip, the compression is provided by a managed-code version of Zlib, included with DotNetZip.
ZipProgressEventArgs	Provides information about the progress of a save, read, or extract operation. This is a base class; you will probably use one of the classes derived from this one.
ZipProgressEventType	In an EventArgs type, indicates which sort of progress event is being reported.

5. Referencias

- DotNetZip – Zip file manipulation in .NET languages – último acceso: 29/06/09
  http://www.codeplex.com/DotNetZip
- Ejemplos en CodeProject – último acceso: 29/06/09
  http://www.codeproject.com
- How to: Enable Streaming – último acceso: 29/06/09
  http://msdn.microsoft.com/en-us/library/ms789010.aspx
- Large Data and Streaming – último acceso: 29/06/09
  http://msdn.microsoft.com/en-us/library/ms733742.aspx
- WCF and large messages – último acceso: 29/06/09
  http://www.haveyougotwoods.com/archive/2008/03/29/wcf-and-large-messages.aspx
- WCF Endpoints: Addresses, Bindings, and Contracts – último acceso: 29/06/09
  http://msdn.microsoft.com/en-us/library/ms733107.aspx

- WCF Streaming: Upload files over HTTP – último acceso: 29/06/09
  http://kjellsj.blogspot.com/2007/02/wcf-streaming-upload-files-over-http.html
- Windows Communication Foundation – último acceso: 29/06/09
  http://msdn.microsoft.com/en-us/library/ms735119.aspx

6. ChatMessage

A continuación se describirán los objetivos propuestos para el desarrollo de la presente biblioteca, así como también sus generalidades e implementación.

1. Objetivo

Se trata de una librería creada especialmente para el manejo de la información ingresada en la sala de chat. Tiene como fin la creación de un archivo con extensión .xml.

2. Generalidades

El archivo creado será adjuntado con los demás archivos que conforman la entrevista, los que se subirán al servidor cuando una entrevista se de por finalizada.

El archivo se construye mediante la creación de un nodo raíz del XML al que, por cada mensaje nuevo ingresado se crea un nodo hijo.

3. Implementación

Para su desarrollo se implementó dos clases una clase *RootFile* encargada de crear el nodo XML y una clase *InterviewMessage* que representará cada nodo hijo ingresado al XML. Este nodo no solamente contiene el texto que el usuario ingresa sino también información interna necesaria para su manejo posterior cuando sea consumida la entrevista a demanda.

Para esta situación es necesario saber el momento preciso en que el mensaje fue ingresado (hora, minutos, segundos) y así sincronizarlo con el video que se está emitiendo.

Se maneja además dos estructuras de datos: una cola *listInterviewMessages* donde se van 'encolando' los mensajes a medida que ingresan a la sala de chat y un array: *arrInterviewMessages* donde, una vez finalizada la entrevista se cargará con los datos que se encuentran en la cola, es de esa manera que a partir del array generado se crea el archivo con extensión .xml deseado.

Se despliegan a continuación la implementación correspondiente:

```
            //Nodo raiz del xml.
            [XmlRootAttribute]
             public class RootFile
             {     //Nodos hijos que representan cada mensaje.
                    [XmlArrayAttribute]
                    public InterviewMessage[] arrInterviewMessages;

             }

      public class InterviewMessage
             {
              public string UserId;
              public string Message;
              public string InterviewId;
              public int  hh;
              public int  mm;
              public int  ss;
             }
```

Código 1: Clases y declaración de array para creación de archivo XML.

Como se describió anteriormente se utiliza una estructura de 'cola' para insertar cada mensaje que ingresa a la sala de chat.

```
 Queue<InterviewMessage> listInterviewMessages
=
        new Queue<InterviewMessage>();
```

Código 2: Estructura de cola utilizada.

Se expone dos métodos:

AddMessage invocado cada vez que un mensaje es ingresado a la sala, carga los campos correspondientes e inserta el mensaje a la cola.

```
public void AddMessage(string strUserId, string strMessage, string
strInterviewId, int hora, int min,int seg )
{
       //Mensaje.
        InterviewMessage objInterviewMessage=new InterviewMessage();
                        objInterviewMessage.UserId=strUserId;
                        objInterviewMessage.Message=strMessage;
                        objInterviewMessage.InterviewId = strInterviewId;
                        objInterviewMessage.hh = hora ;
                        objInterviewMessage.mm = min;
                        objInterviewMessage.ss = seg;
    //Arma cola de mensajes.
        listInterviewMessages.Enqueue(objInterviewMessage);
}
```

3: Implementación método AddMessage.

*Save :*invocado cuando el usuario emisor da por finalizado el chat, copia los mensajes de la cola al array el que es utilizado luego para asociar los mensajes al documento raiz del XML. Finalmente serializa y graba el archivo concluyendo así su creación.

Se describe a continuación el código que implementa el método *Save.*

```
public void Save(string strFileName)
{
       //Archivo a grabar.
        TextWriter writer = new StreamWriter(strFileName);
        InterviewMessage[] arrInterviewMessages= new
           InterviewMessage[listInterviewMessages.Count*2];
   //copia la cola de msj a un array
        listInterviewMessages.CopyTo(arrInterviewMessages,
        listInterviewMessages.Count);
   //Asocia mensajes al documento raiz.
    objRootFile.arrInterviewMessages=arrInterviewMessages;
      //Serializa y salva el archivo.
    serializer.Serialize(writer, objRootFile);
    writer.Close();
}
```

Código 4: Implementación método Save.

7. Servicios dúplex

1. Concepto

Es un tipo de servicio WCF (ver Anexo 2-Windows Communication Foundation), basado en un contrato de servicio 'dúplex', se trata de un modelo de intercambio de mensajes en el que ambos extremos pueden enviar mensajes al otro de manera independiente.

2. Generalidades

La comunicación dúplex se produce cuando un cliente se conecta a un servicio y le proporciona un canal en el que el servicio puede devolver los mensajes al cliente.

3. Contrato dúplex

Se define con un par de interfaces, una interfaz principal del cliente al servicio, y una interfaz de devolución de llamada del servicio al cliente.

La primera es la interfaz de contrato de servicio, que describe las operaciones que un cliente puede invocar.

Ese contrato deberá especificar un *contrato de devolución de llamadas* en la propiedad *System.ServiceModel.ServiceContractAttribute.CallbackContract* el cual define las operaciones a las que el servicio puede llamar en el extremo del cliente.

Un contrato dúplex requiere una sesión, porque se debe establecer un contexto para manejar el conjunto de mensajes que se envían entre el cliente y el servicio.

4. Creación del servicio para Cámara de Gesell

La creación del servicio comienza por la declaración de las diversas interfaces a usar que luego serán implementadas por el propio servicio.

4.1 Declaración de las interfaces

Como se trata de un servicio dual requiere ciertos parámetros más específicos en la declaración del servicio; Además, también necesita la declaración de una interfaz *callback* que será la implementada por el cliente.

```
[ServiceContract(CallbackContract = typeof(IService1Callback), SessionMode =
SessionMode.Required)]
  public interface IService1
  {
   [OperationContract(IsInitiating = true, IsOneWay = true)]
   void Connect(string name,string msg, string id, int hh, int mm, int ss);


   [OperationContract(IsOneWay = true)]
      void SayToServer(string name, string msg, string id, int hh, int mm,
                              int ss);

   [OperationContract(IsTerminating = true, IsOneWay = true)]
   void Disconnect(string name, string msg, string id, int hh, int mm, int
                       ss);

  }
  public interface IService1Callback
  {
    [OperationContract(IsOneWay = true)]
    void SayToClient(string name,string msg,string ident,int hh,int mm,int
                            ss,string []array_users,int tope );
  }
```

Código 1: Declaración ServiceContract con *callback.*

En este caso se puede observar que en la declaración del *ServiceContract* se especifica la interfaz que se utilizará a modo de *callback*, también se ha definido que el servicio requiere sesión por lo que para cada cliente que se conecte se establecerá una sesión y por lo tanto se mantendrá un comportamiento específico para él.

El hecho de sesiones habilitadas implica que debe de haber, al menos un método que tenga *IsInitiating* con valor verdadero y al menos otro *IsTerminating* con valor verdadero, esto significa que serán los primeros (crean la sesión) y últimos (destruyen la sesión) en ser utilizados respectivamente.

4.2 Implementación del servicio

Luego de la creación de las interfaces comienza su implementación

```
public class Service1 : IService1
{……
}
```

Código 2: Implementación de Interfaces.

Por tratarse de un servicio que utilizará la infraestructura Internet Information Service (IIS), se crea el archivo *svc* que será referenciado en el servidor Web desde un directorio virtual.

```
<%@ ServiceHost Language="C#" Debug="true" Service="Web_Ws_Chat.Service1"
CodeBehind="Service1.svc.cs" %>
```

Código 3: Archivo *svc* WCF.

Finalmente, en el archivo de configuración se especifica los servicios a publicar:

```
<configuration>
   <system.serviceModel>
   <bindings>
   <wsDualHttpBinding>
   <binding name="WsDPlain">
    <security mode="None">
     <message clientCredentialType="None"
          algorithmSuite="Default"
          negotiateServiceCredential="false"/>
    </security>
   </binding>
  </wsDualHttpBinding>
  </bindings>
   <services>
   <service behaviorConfiguration="HTTPGET"
        name="Web_Ws_Chat.Service1">
      <endpoint address="Service1"
        binding="wsDualHttpBinding"
        bindingConfiguration="WsDPlain"
        bindingNamespace="http://Web_Ws_Chat/2009/09"
        contract="Web_Ws_Chat.IService1" />
      <endpoint address="mex"
        binding="mexHttpBinding"
        contract="IMetadataExchange" />
   </service>
   </services>
  <behaviors>
   <serviceBehaviors>
    <behavior name="HTTPGET">
     <serviceDebug includeExceptionDetailInFaults="false" />
     <serviceMetadata httpGetEnabled="true" />
     <serviceThrottling maxConcurrentCalls="640"maxConcurrentSessions="500"
      maxConcurrentInstances="1000" />
    </behavior>
   </serviceBehaviors>
  </behaviors>
</system.serviceModel>
      </configuration>
```

Código 4: Archivo de configuración del servicio WCF.

4.3 Generación de proxies

Existen dos maneras de generar las clases proxies que se invocarán desde los clientes.

Una, es mediante una herramienta llamada *svcutil* a la que se indica la dirección del archivo *svc*, seguido de otros parámetros de configuración.

Y otra, es mediante el uso de la plantilla de Windows Presentation Foundation (WPF) que la genera automáticamente cuando se agrega la referencia al servicio mediante la opción *Add Service References*, fue la utilizada en el proyecto.

Figura 1: Generación proxy de un servicio *svc.*

4.4 Invocación del servicio desde la aplicación cliente

Se utiliza la clase cliente que se ha generado:

```
InstanceContext context = new InstanceContext(this);
proxy = new Integracion.Servicios.Service1Client(context);
```

Código 5: Uso de proxy WCF en aplicación cliente.

Luego se implementa la interfaz *callback* por tratarse de un servicio dúplex (explicado anteriormente)

```
public partial class WinPcpal : Servicios.IService1Callback
        {……}
```

Código 6: Implementación *callback* de un servicio WCF.

A continuación se despliega el archivo de configuración generado en la aplicación cliente luego de agregar la referencia al servicio:

```
<configuration>
  <system.serviceModel>
  <bindings>
   <wsDualHttpBinding>
     <binding name="WSDualHttpBinding_IService1"closeTimeout="00:01:00"
                    openTimeout="00:01:00" receiveTimeout="00:10:00"
              sendTimeout="00:01:00"
                  bypassProxyOnLocal="false" transactionFlow="false"
                  hostNameComparisonMode="StrongWildcard"
           maxBufferPoolSize="524288" maxReceivedMessageSize="65536"
              messageEncoding="Text"
              textEncoding="utf-8"  useDefaultWebProxy="true">
          <readerQuotas maxDepth="32" maxStringContentLength="8192"
                    maxArrayLength="16384"
                    maxBytesPerRead="4096" maxNameTableCharCount="16384" />
          <reliableSession ordered="true"
           inactivityTimeout="00:10:00"/>
          <security mode="None">
           <message clientCredentialType="Windows"
                     negotiateServiceCredential="true" />
          </security>
       </binding>
    </wsDualHttpBinding>
   </bindings>
   <client>
      <endpoint
             address="http://Servidor_IIS_Servicios/Chat/Service1.svc/Service1"
             binding="wsDualHttpBinding"
             bindingConfiguration="WSDualHttpBinding_IService1"
      contract="Servicios.IService1"
             name="WSDualHttpBinding_IService1"/>
    </client>
  </system.serviceModel>
</configuration>
```

Código 7: Archivo de configuración de aplicación cliente WCF.

5. Referencias

- Aplicación de Servicios dúplex – último acceso: 29/06/09
  http://msdn.microsoft.com/es-es/library/ms731064.aspx

8. Chat del canal del mismo nivel

1. Concepto

Al mencionar canal del mismo nivel se habla de *PeerChannel,* es una tecnología de comunicación punto a punto (P2P) entre varias partes de Windows Communication Foundation (WCF). Proporciona un canal de comunicación P2P basado en mensajes seguros, escalables y confiables. Habilita la distribución de contenido, la colaboración P2P y el equilibrio de la carga, entre otros.

2. Aplicación

Un buen uso de esta tecnología es el chat, donde un grupo de personas conversan entre ellas de igual a igual, sin necesidad de servidores.

3. Generalidades

No se basa en el concepto cliente y servicio, se trata de una aplicación real punto a punto, en la que cada instancia actúa como par de otras instancias, es decir cada instancia puede enviar mensajes a otras así como recibir mensajes de ellas mediante contracto dúplex.

4. Terminologías

Se introducen nuevos conceptos al trabajar con esta tecnología, ellos son:

Red

Se trata de una colección con nombre de nodos del mismo nivel que se comunican entre sí y que se identifican de forma única por un identificador de red.

Los nodos activos en la red publican su nombre de manera que otros puedan ubicarlos. Una red se caracteriza por su conectividad elástica en un entorno en que los nodos se combinan constantemente abandonando la red, cambiando su pertenencia y optimizándose dinámicamente según los modelos de tráfico.

Nodo del mismo nivel

Se trata de nodos pertenecientes al extremo de una red, puede pasar que en una única aplicación existan varios nodos de un mismo nivel pertenecientes a redes diferentes.

Resolución del mismo nivel

Es quien resuelve un identificador de red en las direcciones de extremo de los nodos de la red. Es la manera de propagar los mensajes por ella cuando un nodo del mismo nivel utiliza esas direcciones para conectarse a otros nodos en la red.

5. Implementación

Para llevar a cabo su implementación hay que tener clara la manera en que trabaja los canales en un mismo nivel, alguno de esos puntos son los siguientes:

- Cada instancia del chat debe utilizar la misma dirección de punto final para asegurar de esa manera que los mensajes enviados por una, lo reciben todas las demás instancias.
- Las instancias pueden encontrarse entre sí mediante una resolución personalizada o a través de la resolución del mismo nivel predeterminado (PNRP).
- En Windows Server 2003 no está disponible PNRP.

6. Referencias

- Chat de canal del mismo nivel – último acceso: 30/06/09
  Versión ingles: http://msdn.microsoft.com/en-us/library/ms751502.aspx
  Versión español: http://msdn.microsoft.com/es-es/library/ms751502.aspx
- Conexiones de redes punto a punto – último acceso: 30/06/09
  http://msdn.microsoft.com/es-es/library/ms733761.aspx

A3 – Anexos referidos a Base de Datos

1. Base de Datos

Se decide usar el sistema de gestión de base de datos relacionales Microsoft Sql Server 2005 Express aprovechando la fuerte integración con Visual Studio que tiene. Desde el mismo entorno de Visual Studio es posible acceder a la base de datos y realizar modificaciones sobre las estructuras de las bases, al igual que sobre los datos. La herramienta se instala junto con Visual Studio.

Sql Server Express permite la creación de sistemas sólidos y fiables debido a la robustez del sistema de base de datos. Cuenta con un optimizador avanzado de consultas, importante para resolver en tiempos mínimos las consultas, tratando de reducir el tiempo de respuesta de la aplicación remota. Maneja diferentes niveles de seguridad para el acceso a los datos, contando entre otros con compatibilidad con la autenticación de Windows, usada en el proyecto.

Pero por ser una versión gratuita, es una versión con limitaciones. En caso de necesitar ampliar las funcionalidades del sistema, puede ser actualizable a otras versiones. De esta manera podrían dejarse de lado problemas que pudieran surgir con: la cantidad de información en la base, se cuenta con un máximo de 4 GB; lentitud de respuesta asociada a la cantidad de conexiones concurrentes; falta de interfaz gráfica para la administración del servidor de base de datos, entre otros.

La limitante de no tener interfaz gráfica para la administración es resuelta con la herramienta Management Studio Express Edition que también es gratuita y permite administrar la base de datos en forma gráfica. A través de esta herramienta se pueden crear o eliminar bases de datos y sus tablas y vistas. Se puede visualizar la información de las tablas y modificar los datos. Además se puede administrar la seguridad del servidor y de las bases de datos existentes.

1. Entidades

La base de datos *Gesell* creada en Sql Server 2005 Express mantiene los datos del sistema.

Las entidades que se representan en las tablas relacionadas a los usuarios son:

- Users: Usuarios del sistema.
- Specialities: Especialidades definidas en el sistema. Se asignan las especialidades a los usuarios.

Las entidades necesarias para representar las publicaciones y las entrevistas son:

- Publications: Publicaciones hechas en el sistema.
- Interviews: Entrevistas administradas por el sistema.
- MetadataInfo: Información definida para la metadata de las entrevistas.

Para usar en forma auxiliar en el momento de emisión en vivo de una entrevista, también se tienen representadas las siguientes entidades para el funcionamiento del Chat:

- ChatUsers: Usuarios conectados al Chat de las entrevistas.
- ChatMessage: Mensajes enviados a través del Chat de las entrevistas por los usuarios.

Otros datos necesarios para el sistema se representan a través de las entidades:

- Entities: Entidades usadas en el sistema.
- Configuration: Configuraciones que se definen para la ejecución del sistema.
- Metadata: Metadata asociada a las entrevistas.

También se deben representar relaciones entre entidades. Estas son:

- UsersSpecialities: Relaciones entre usuarios y especialidades.
- InterviewsGuests: Relaciones entre entrevistas y usuarios invitados.

2. Tablas

Las tablas creadas en la base de datos son:

2.1 Users

	Nombre	Tipo	Descripción
	UserId	Alfanumérico (20)	Identificador de usuario.
	UserName	Alfanumérico (40)	Nombre(s) del usuario.
	UserSecondName	Alfanumérico (40)	Apellido(s) del usuario.
	Password	Alfanumérico (40)	Contraseña del usuario.
	Email	Alfanumérico (40)	Correo electrónico del usuario.
	DateInserted	Fecha Hora	Fecha en que se dió de alta al usuario en el sistema.
	LastVisit	Fecha Hora	Última visita del usuario.
	ProfileId	Numérico Entero	Identificador de perfil de usuario. (*)
	ProfileEntity	Alfanumérico (20)	Entidad perfil. (*)

(*) Los perfiles de usuario se definen como entidades en la tabla Entities. La entidad para el perfil de usuario es 'Profile'. Los valores posibles para perfil de usuario son:

1- Administrador

2- Usuario Común

2.2 Specialities

	Nombre	Tipo	Descripción
	SpecialityId	Numérico Entero	Identificador de especialidad.
	SpecialityDesc	Alfanumérico (40)	Descripción de especialidad.

2.3 Publications

	Nombre	Tipo	Descripción
	PublicationId	Numérico Entero	Identificador de publicación.
	Type	Alfanumérico (1)	Tipo de publicación. (*)
	Title	Alfanumérico (200)	Título de la publicación ó enlace.
	Comment	Alfanumérico (1000)	Comentarios
	Link	Alfanumérico (200)	En el caso de publicación de tipo publicación, es el link al documento publicado. En el caso de publicación de tipo enlace, es el enlace al sitio publicado.
	Authors	Alfanumérico (200)	En el caso de publicación de tipo publicación, son los autores de la publicación. En el caso de publicación de tipo enlace, este campo no se usa.
	UserId	Alfanumérico (20)	Identificador de usuario que realiza la publicación.

	DateInserted	Fecha Hora	Momento en que se da de alta la publicación.

Los valores posibles para tipo de publicación son:

P- Publicaciones

E- Enlaces

2.4 Interviews

	Nombre	Tipo	Descripción
	InterviewId	Numérico Entero	Identificador de la entrevista.
	InterviewDesc	Alfanumérico (40)	Descripción de la entrevista.
	DateAndTime	Fecha Hora	Fecha y hora de realizada la entrevista.
	StatusId	Numérico	Estado de la entrevista. (*)
	StatusEntity	Alfanumérico (20)	Entidad estado. (*)
	FileSize	Numérico Entero	Tamaño del archivo expresado en kbytes.
	FileLength	Numérico Entero	Largo del archivo expresado en segundos.
	QtyPositiveVote	Numérico Entero	Cantidad de votos positivos sobre la entrevista.
	QtyNegativeVote	Numérico Entero	Cantidad de votos negativos sobre la entrevista.
	QtyDownload	Numérico Entero	Cantidad de descargas de la entrevista.
	SpecialityId	Numérico Entero	Identificador de especialidad.
	UserId	Alfanumérico (20)	Identificador de usuario.

(*) Los estados de una entrevista se definen como entidades en la tabla Entities. La entidad para el estado es 'Status'. Los valores posibles para los estados son:

1- Definir Portada.

2- Agendar.

3- Emitir.

4- Cerrar.

5- Publicar.

6- Finalizada.

2.5 MetadataInfo

	Nombre	Tipo	Descripción
	InterviewId	Numérico Entero	Identificador de la entrevista.
	Fieldád	Numérico Entero	Identificador del campo.
	Data	Alfanumérico (1000)	Información.

La información asociada a cada entrevista depende de la definición de metadata que se haga para la especialidad asociada a la entrevista. En esta tabla se mantiene la información referente a los campos configurables definidos en la metadata de la especialidad. A diferencia de los campos incluidos en la tabla de entrevistas (Interviews), la información que se mantiene en esta tabla, es configurable a través del sistema. Los campos de Interviews, son fijos para todas las entrevistas. Igualmente, los datos aquí definidos junto con los datos definidos en la tabla Interviews, conforman toda la información respecto de una entrevista.

2.6 ChatUsers

	Nombre	Tipo	Descripción
	InterviewId	Numérico Entero	Identificador de la entrevista.
	UserId	Alfanumérico (20)	Identificador de usuario.
	DateAndTime	Fecha Hora	Fecha y hora en que se conecta el usuario a la emisión de la entrevista.

2.7 ChatMessage

	Nombre	Tipo	Descripción
	InterviewId	Numérico Entero	Identificador de la entrevista.
	UserId	Alfanumérico (20)	Identificador de usuario.
	MessageId	Numérico Entero	Identificador del mensaje.
	Message	Alfanumérico (200)	Mensaje enviado a la entrevista por el usuario.
	DateAndTime	Fecha Hora	Fecha y hora en que se recibe el mensaje en el servidor.

2.8 Entities

	Nombre	Tipo	Descripción
	Entity	Alfanumérico (20)	Entidad a la que pertenece el valor.
	Id	Numérico Entero	Valor.
	Description	Alfanumérico (20)	Descripción.

En esta tabla se permiten definir grupos de entidades y sus valores específicos. Estas entidades se usan en diferentes partes del sistema y consisten de un valor y una descripción. La descripción se usa en las interfaces de usuario del sistema. Los valores son los asociados que se incluyen en otras tablas.

Las entidades definidas en esta tabla son:

- Perfiles de usuario, 'Profile'.
- Estados de las entrevistas, 'Status'.
- Tipos de las publicaciones, 'Type'.

Cuando en otras tablas se utilizan entidades, se definen 2 campos para establecer una entidad. En el caso de los Usuarios, los campos ProfileId y ProfileEntity; para las Entrevistas, los campos StatusId y StatusEntity; para Metadata los campos TypeId y TypeEntity. Aunque los campos ProfileEntity, StatusEntity y TypeEntity tengan valores prefijados y todos los registros de cada tabla tengan igual valor, se realiza de esta manera para obtener un mejor funcionamiento al utilizar las relaciones en Linq to Sql. Ver anexo Anexo - Acceso a Datos.

2.9 Configuration

	Nombre	Tipo	Descripción
	Parameter	Alfanumérico (20)	Parámetro a configurar.
	Value	Alfanumérico (200)	Valor asociado.

La idea es que la única configuración necesaria y disponible del lado del cliente sean las URLs a donde se han publicado los servicios Web. Esto queda configurado en el archivo Web.config del lado del cliente. El resto de las configuraciones necesarias en el sistema se mantienen en la tabla de configuraciones. Se hace de esta manera por seguridad y para centralizar los posibles cambios asociados a infraestructura.

Los parámetros que se configuran en esta tabla son:

- FolderPreview: Path absoluto al directorio en el servidor de archivos donde se almacenan las imágenes de vista previa de las entrevistas a ser mostradas en la interfaz de usuario.
- StreamingServer: Dirección IP del servidor de streaming.
- ServerUpload: camino absoluto al directorio en el servidor de archivos donde se almacena el contenido de las entrevistas publicadas.

2.10 Metadata

	Nombre	Tipo	Descripción
	FieldId	Numérico Entero	Identificador de campo.
	SpecialityId	Numérico Entero	Identificador de especialidad.
	FieldDesc	Alfanumérico (40)	Descripción del campo visible en la interfaz de usuario. (*2)
	MaxLength	Numérico Entero	Largo máximo del campo. (*2)
	InUse	Booleano	Indica si el campo está en uso. (*3)
	TypeId	Numérico Entero	Identificador de tipo de campo. (*1)
	TypeEntity	Alfanumérico (20)	Entidad tipo de campo. (*1)
	Position	Numérico Entero	Posición del campo, establece el orden en que se visualizan los campos.

(*1) Los tipos de campo se definen como entidades en la tabla Entities. La entidad es 'Type' y permite indicar si un campo se usa en la portada de la entrevista ó en el cierre de la misma. Los valores posibles son:

1- Portada

2- Cierre

(*2) Se permite configurar tanto la descripción del campo que se muestra en la interfaz de usuario, como el largo máximo permitido para el ingreso de información en dicho campo.

(*3) Si en la configuración del sistema decide dejar de usarse un campo ya existente, puede darse que alguna especialidad ya tenga información asociada a dicho campo. Esta información se sigue usando. En caso de que un campo se establezca como que no está más en uso, no se visualizará en la interfaz para futuras entrevistas, pero si para las pasadas.

2.11 UsersSpecialities

	Nombre	Tipo	Descripción
	UserId	Alfanumérico (20)	Identificador de usuario.
	SpecialityId	Numérico Entero	Identificador de especialidad.

Mantiene las especialidades asociadas a cada usuario.

2.12 InterviewsGuests

	Nombre	Tipo	Descripción
	UserId	Alfanumérico (20)	Identificador de usuario.
	InterviewId	Numérico Entero	Identificador de entrevista.

Mantiene los usuarios invitados a cada entrevista.

3. Referencias

- Microsoft SQL Server 2005 Express Edition – último acceso: 30/06/09
  http://www.microsoft.com/downloads/details.aspx?familyid=220549B5-0B07-4448-8848-DCC397514B41&displaylang=es
- Microsoft SQL Server Management Studio Express – último acceso: 30/06/09
  http://www.microsoft.com/downloadS/details.aspx?familyid=C243A5AE-4BD1-4E3D-94B8-5A0F62BF7796&displaylang=es

2. Acceso a datos

1. Arquitectura

En el proyecto se utiliza la siguiente arquitectura para solucionar el acceso a datos. La misma consiste en dividir la capa de acceso a datos en tres subcapas:

- Contexto de datos de Linq to Sql.
- Entidades.
- Servicios de WCF.

Todos los componentes de esta capa están instalados en el servidor de aplicaciones. En el servidor se expone solamente lo necesario a través de la subcapa de 'Servicios de WCF'. Los datos a los clientes se envían serializados, como representaciones XML de las entidades o como objetos proxy. En la subcapa de servicios, también se agrega lógica de negocios. Los datos que se envían al cliente se obtienen a partir de las clases entidades, que no son 'visibles' desde la aplicación cliente remota. Es en estas clases y a través del contexto de datos de Linq to Sql que se accede a los datos de la base.

Se muestra en la siguiente imagen la arquitectura propuesta.

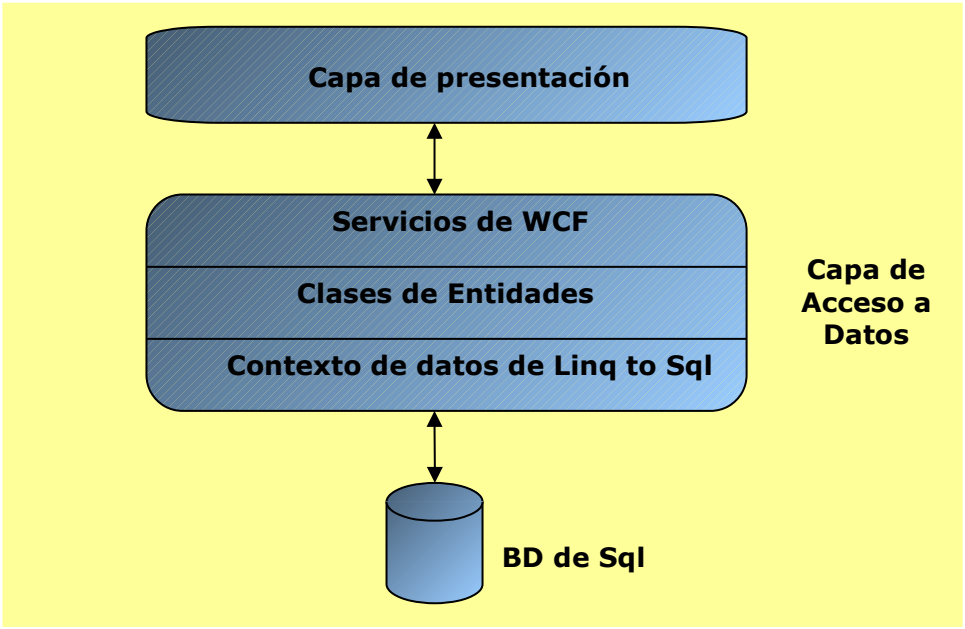


Figura 1: Arquitectura de acceso a datos.

2. Linq to Sql

Linq to Sql es un componente del Framework 3.5 de .Net. Se ven a continuación algunas de sus características importantes.

2.1 Modelo

Linq to Sql permite administrar los datos relacionales como objetos, asigna al modelo de base de datos un modelo de objetos. En tiempo de ejecución, Linq to Sql resuelve las consultas Sql a ser enviadas a la base de datos a partir del modelo de objetos. La aplicación utiliza la API de Linq to Sql para solicitar la ejecución de la consulta. El proveedor de Linq to Sql transforma la consulta en texto Sql. La consulta Sql es ejecutada por el proveedor de acceso a datos ADO.

El proveedor ADO es un conjunto de componentes del Framework de .Net que permite acceder y modificar datos tanto de bases relacionales como de no relacionales. Algunas clases de utilidad de este componente pueden ser: *Connection*, que se usa para establecer la conexión con la base de datos; *Command*, usado para realizar alguna operación sobre los datos de la base de datos; *DataReader*, utilizada para procesar un conjunto de registros, uno a la vez.

Cuando el proveedor ADO ejecuta la consulta sobre la base de datos devuelve los resultados en un DataReader. El proveedor Linq to Sql es el encargado de convertirlos en una colección de objetos de usuario. De esta manera, los datos de la base de datos quedan mapeados al modelo de objetos.

En la imagen siguiente puede verse en la parte superior, el flujo a través del cual se resuelve la consulta en Linq to Sql. Corresponde al modelo de base de datos. En la parte inferior se visualiza el modelo de objetos asociado por Linq to Sql.

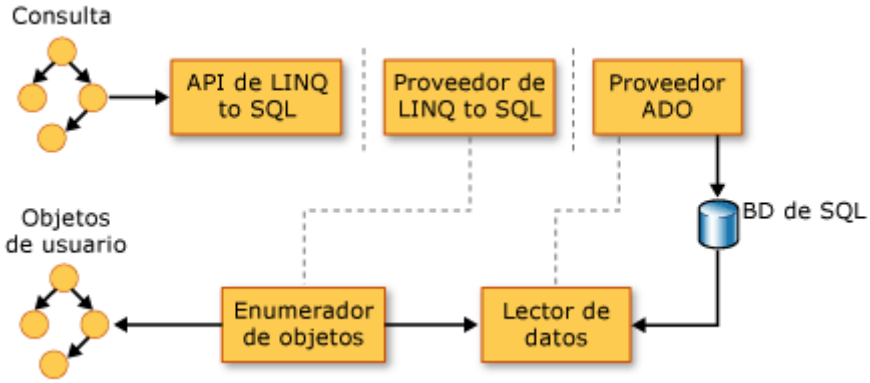


Figura 2: Diagrama de ejecución de la consulta.

2.2 Carga

Al ejecutar una consulta sobre el modelo de datos, puede obtenerse un objeto de una clase. Y solamente se obtiene el objeto solicitado. En el modelo probablemente este objeto tenga otros objetos relacionados. Debido a que el modelo es de entidad-relación, se podrá acceder a partir del objeto a los objetos relacionados de otras clases. Estas relaciones quedan definidas a partir de las claves primarias y claves externas. La ejecución de la consulta adecuada para resolver esta invocación se resuelve de forma invisible al programador. La carga de los objetos que se acceden a partir de las relaciones con otros objetos se realiza por defecto a demanda, es decir, cuando se invoca por código al objeto relacionado o alguna propiedad o método del mismo. A esta forma de carga se le denomina 'carga aplazada'. Se puede también utilizar la 'carga inmediata'. En este caso se resuelve en una única vez la carga de varios objetos de la base, a memoria. No es el caso usado en el proyecto.

2.3 Transaccionalidad

Linq to Sql utiliza el método *SubmitChanges* para ejecutar los cambios sobre la base de datos de los objetos a ser modificados, insertados o eliminados.

Linq to Sql tiene tres modelos de transacciones. Se enumeran en el orden en que se realizan las comprobaciones.

Transacción local explícita.

Cuando se invoca al método *SubmitChanges*, si se tiene una transacción establecida, las sentencias sql a ser ejecutadas se realizan en el contexto de la transacción. Es el programador quien debe confirmar o cancelar la transacción luego de la ejecución de las consultas.

Transacción distribuible explícita.

Si no existe una transacción local definida explícitamente, se comprueba si existe una transacción distribuible explícita. Si es este el caso y se invoca a *SubmitChanges* en el ámbito de esta transacción activa, Linq to Sql lo detecta y no crea una nueva transacción. En este caso no realiza el cierre de la conexión.

Transacción implícita.

Si no se encuentra en ninguno de los casos anteriores, Linq to Sql inicia una transacción local y ejecuta los comandos generados dentro de la misma. Luego confirma la transacción.

En cualquier proyecto es posible encontrar cualquiera de los tres modelos. El programador es quien decide si define una transacción explícita o distribuible explícita, dentro de la cual se ejecutarán una serie de cambios en los datos de la base; o si no define una transacción explícitamente dejando esta labor a Linq to Sql. En cualquiera de los casos deberá tener en cuenta que acciones tomar con la transacción y la conexión explícitamente y que acciones realiza Linq to Sql por sí solo.

2.4 Serialización

La serialización corresponde al procedimiento en el que un objeto es convertido a una secuencia XML. Se serializan las propiedades y campos públicos del objeto y los valores devueltos por los métodos públicos y sus parámetros.

Luego de serializado un objeto, puede ser usado para el transporte entre dos aplicaciones a través de Internet, sin necesidad de que ambas se estén usando en la misma plataforma, al ser XML un estándar abierto. En la aplicación que consume el objeto, se lo puede reconstruir deserializando la secuencia XML.

Cuando un objeto de Linq to Sql se serializa, la entidad se desasocia del contexto de datos. Es por esta razón que cuando se deben hacer modificaciones o eliminaciones de datos, en conexiones remotas, como se verá más adelante; la entidad debe asociarse nuevamente al contexto de datos, luego de que se recibe la secuencia XML y se deserializa el objeto.

Linq to Sql solamente admite la serialización unidireccional. Consiste en marcar la propiedad que se encuentra del lado primario de una relación de clave externa y clave principal. De esta manera se evitan ciclos.

La serialización por si misma, activa la carga aplazada de los objetos.

2.5 Seguridad

Dado que el string de conexión a la base de datos queda disponible en archivos de configuración, como se verá más adelante, es recomendable usar seguridad integrada para realizar una conexión confiable. De esta manera no es necesario almacenar usuario ni contraseña en la configuración.

3. Modelo de datos

Como se mencionó anteriormente, Linq to Sql permite modelar la información de la base de datos como un modelo de objetos relacional. Cuenta para esto con una herramienta incluida en la IDE de Visual Studio llamada 'Diseñador relacional de objetos'. Luego de que el modelo se ha completado es posible obtener código generado automáticamente por Visual Studio, a partir de su definición.

3.1 Diseñador relacional de objetos

El 'Diseñador relacional de objetos' proporciona una interfaz de usuario para trabajar con Linq to Sql. Las tablas de la base de datos origen se mapean con clases llamadas 'clases

entidades', los campos de cada tabla como propiedades de cada clase. Cada vez que se instancia un objeto de una clase, llamado 'entidad', queda mapeado a un registro de la tabla.

En este proyecto se utiliza esta interfaz para incluir las tablas diseñadas en la base de datos de Microsoft Sql Server llamada *Gesell*, que mantiene los datos del sistema.

El uso del diseñador es bastante simple. Dentro del proyecto de Visual Studio creado para resolver el acceso a datos, se agrega un nuevo elemento de tipo 'Clases de Linq to Sql'. Este archivo es de extensión dbml.

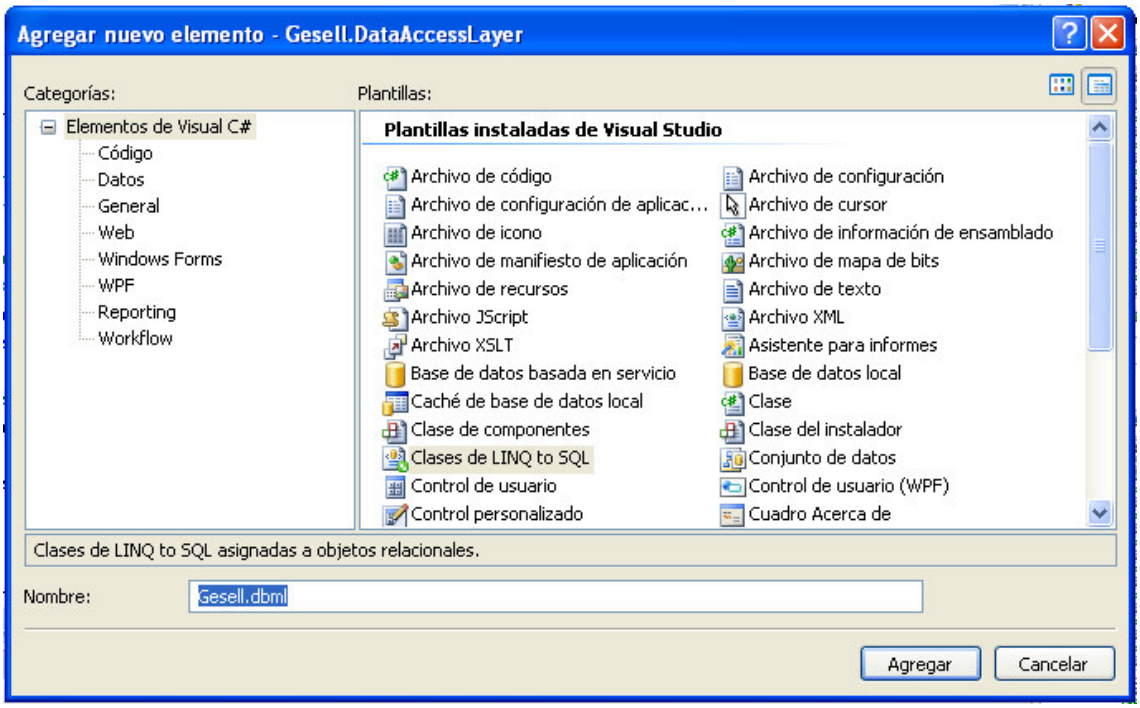


Figura 3: Clases de Linq to Sql.

Luego desde el 'Explorador de servidores', se debe establecer la conexión al origen de datos. En el caso del proyecto, la base de datos de Sql Server, *Gesell*.

Desde la base de datos deben arrastrarse las tablas de la base hacia el diseñador para poder incluir las clases de entidades que se mapean con las mismas.

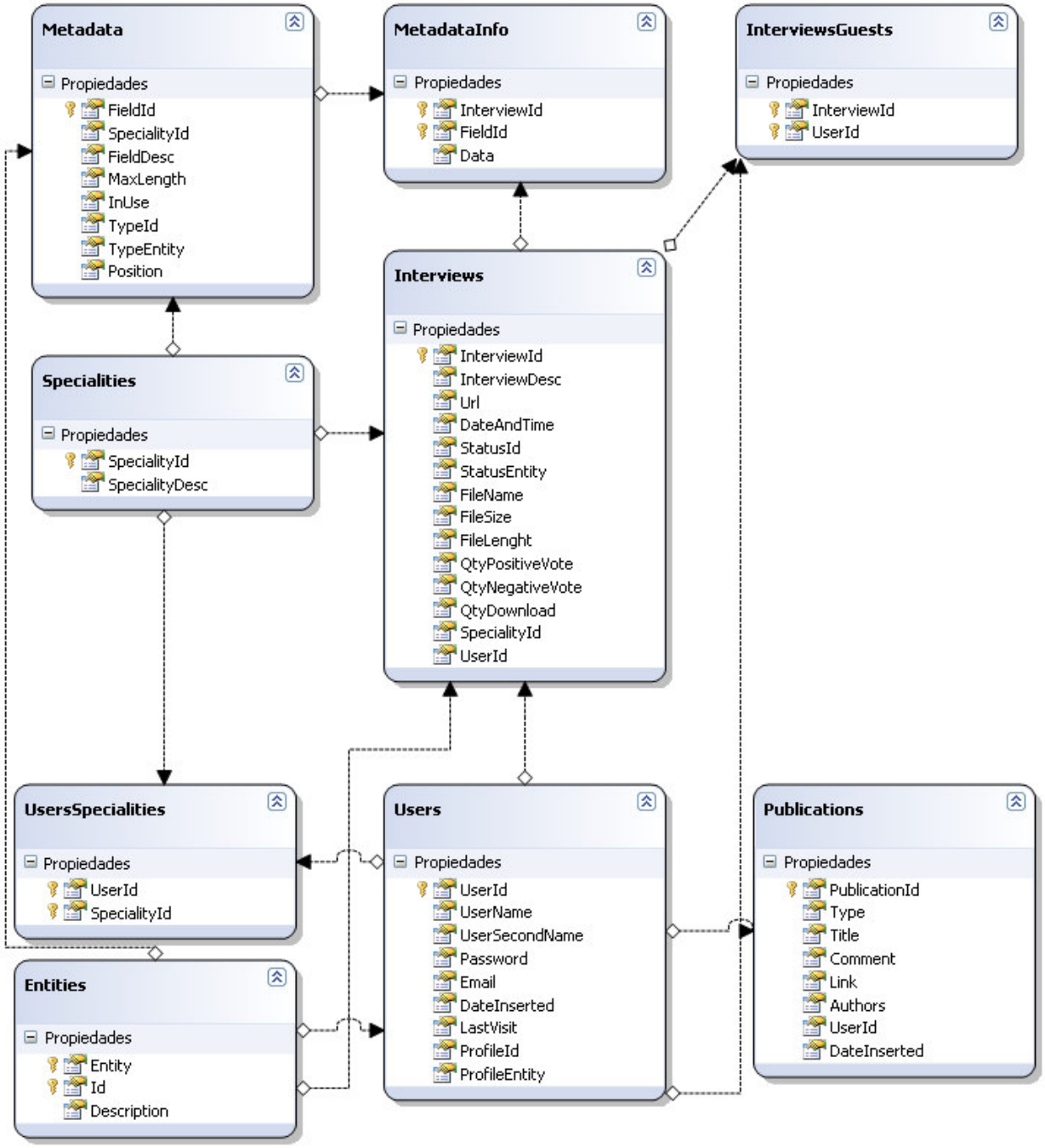


Figura 4: Diseñador relacional de objetos.

Luego de haber incluido todas las tablas, puede visualizarse un diagrama del estilo entidad-relación. Cada tabla se representa como una entidad. Cada relación presente en el diagrama se genera automáticamente a partir de las claves primarias y foráneas de la base de datos. También podrían haberse creado desde el propio diseñador pero hemos preferido crear las estructuras y relaciones en la propia base de datos.

3.2 Generación automática

Al grabar el archivo *.dbml, se tienen tres nuevos archivos en el proyecto:

- *Gesell*.dbml: archivo XML donde se define el mapeo.
- *Gesell*.dbml.layout: archivo XML que usa el diseñador.

- *Gesell*.designer.cs: archivo c sharp donde se generan las clases entidades y la clase de contexto de datos.

Visual Studio genera el código base de las clases entidades, que luego podrán ser extendidas si es necesario. Se genera para cada clase entidad una clase parcial, en donde los campos de cada tabla corresponden a propiedades de cada clase. Además se incluyen métodos de uso habitual y que son necesarios para poder trabajar con las instancias de las clases. Por ejemplo, los métodos que permiten agregar entidades asociadas a alguna propiedad de la instancia o los que definen qué hacer cuando una propiedad cambia su valor.

Se genera también la clase de contexto de datos llamada *GesellDataContext*. Es a través de esta clase que las clases entidades realizan consultas sobre la base de datos. Esta clase contiene propiedades que representan las entidades que se modelan y métodos que representan los procedimientos almacenados de la base de datos. Cuando se crea una instancia de esta clase funciona como punto de entrada a la base de datos definida en la conexión. Luego de recuperar entidades de la base, hace un seguimiento de los cambios que se aplican. Mantiene una 'memoria caché de identidad' que hace que si una entidad se recupera de la base más de una vez dentro del mismo contexto, se utiliza la misma entidad y no genera una nueva.

En el archivo de configuración *app.config* se agrega el string de conexión para acceder a la base de datos.

Código

```
<?xml version="1.0" encoding="utf-8" ?>
<configuration>
  <configSections>
  </configSections>
  <connectionStrings>
    <add name="Gesell.DataAccessLayer.Properties.Settings.
              GesellConnectionString"
              connectionString="Data Source=SERVIDOR-STREAM\ SQLEXPRESS;Initial
              Catalog=Gesell;Integrated Security=True"
       providerName="System.Data.SqlClient" />
  </connectionStrings>
      </configuration>
```

1: Archivo de configuración.

4. Extensión del modelo de datos

Las clases parciales generadas automáticamente a partir del modelo de datos tienen los elementos básicos de una clase. Es tarea del programador extenderlas para proveer a las clases de todas las funcionalidades necesarias según los requerimientos del sistema del que forma parte.

4.1 Clases parciales

Las clases parciales generadas se extienden mediante otras clases parciales. Se habla de 'clases parciales' porque una clase puede definirse en diferentes lugares en forma parcial. Pero el conjunto de estas clases parciales definen una única clase.

Se deben definir las clases parciales que extienden las generadas por Linq to Sql en archivos diferentes. Si hubiera una modificación en la base de datos y se actualizara el modelo de datos, al volver a generar el código, se perdería lo programado, si se hubiera desarrollado en el mismo archivo.

Para cada entidad que se extiende se crea un nuevo archivo *.cs. Como ejemplo, la clase parcial extendida para la entidad Usuario se encuentra en el archivo *DALUsers.cs*. La clase parcial generada por Linq to Sql se encuentra dentro de *Gesell.dbml*, junto con el resto de las clases parciales generadas automáticamente. Todos estos archivos se encuentran dentro del proyecto *Gesell.AccessLayer*.

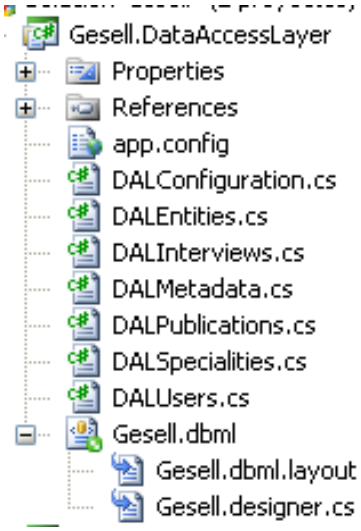


Figura 5: Archivos del componente Gesell.DataAccessLayer.

Se indica que una clase es parcial de la siguiente manera:

```
namespace Gesell.DataAccessLayer
{
  //Clase parcial que complementa la definida por Linq to Sql.
  public partial class Users
  {
    const string PROFILE="Profile";
```

Código 2: Clase parcial.

4.2 Métodos

Básicamente, se extienden las diferentes clases con los siguientes métodos:

- Load: Para obtener datos de las entidades desde la base de datos.
- Save: Para realizar altas o modificaciones de entidades.
- Delete: Para eliminar entidades de la base de datos.

En la mayoría de las clases se tienen también sobrecarga de estos métodos, según las necesidades del sistema. Se optó por tener varios métodos con igual firma dentro de una clase para tareas muy similares, porque aportan claridad a las clases.

Puede verse en la figura los métodos definidos para la clase entidad *Interviews*. Algunos corresponden a los generados automáticamente por Linq to Sql y otros a los definidos en la extensión de la clase.

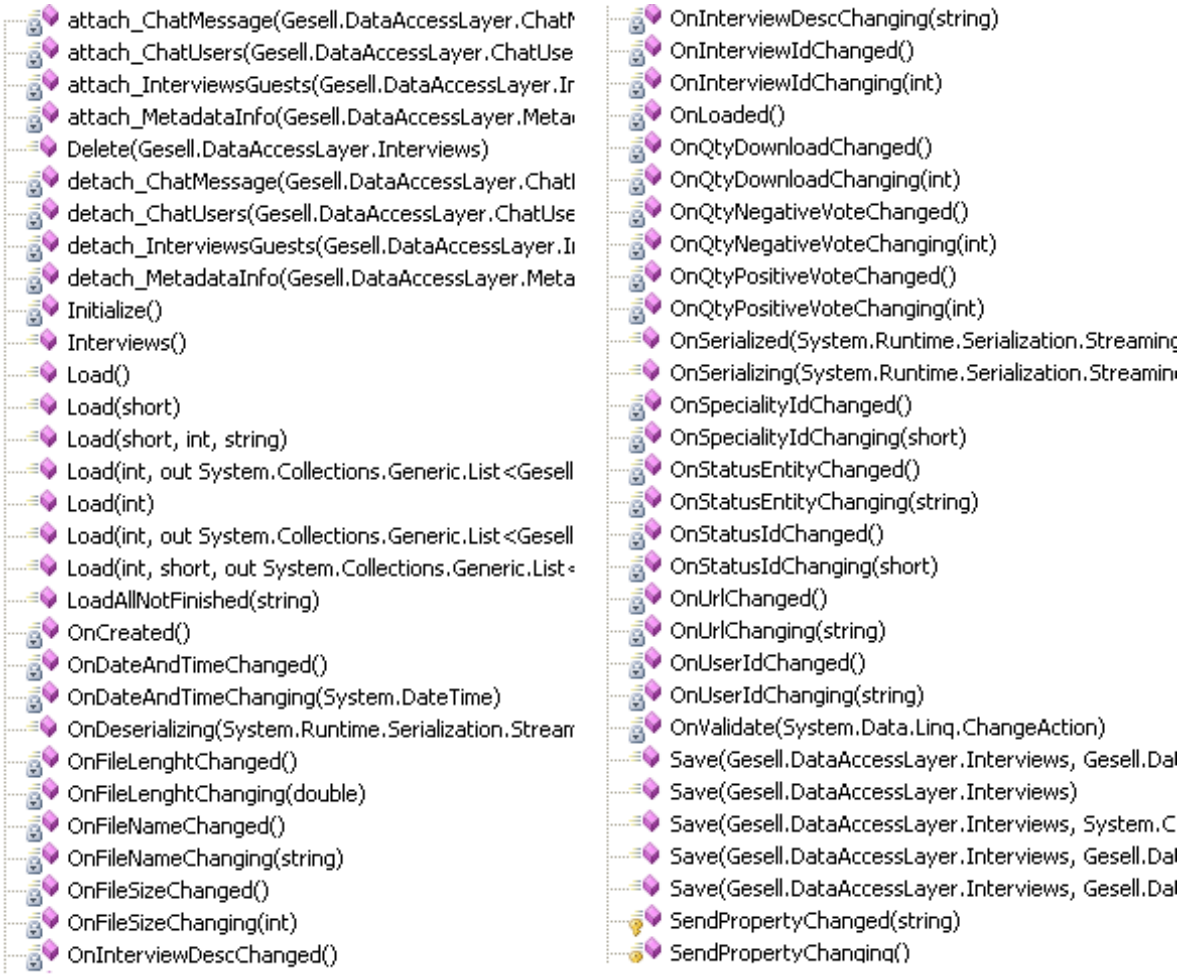


Figura 6: Métodos de clase entidad.

5. Subcapas de acceso a datos

Como se mencionó anteriormente, para solucionar el acceso a datos, se divide la capa en tres subcapas:

- contexto de datos de Linq to Sql.
- Entidades.
- servicios de WCF.

La clase *GesellDataContext* conforma la subcapa de contexto de datos de Linq to Sql.

Las clases parciales generadas automáticamente a partir del 'Diseñador relacional de objetos', y las clases parciales desarrolladas para extender las funcionalidades de estas clases, conforman las subcapas de entidades.

Estas dos subcapas se incluyen dentro del componente *Gesell.DataAccessLayer*. En la siguiente figura puede verse la totalidad de las clases que se incluyen en el componente.

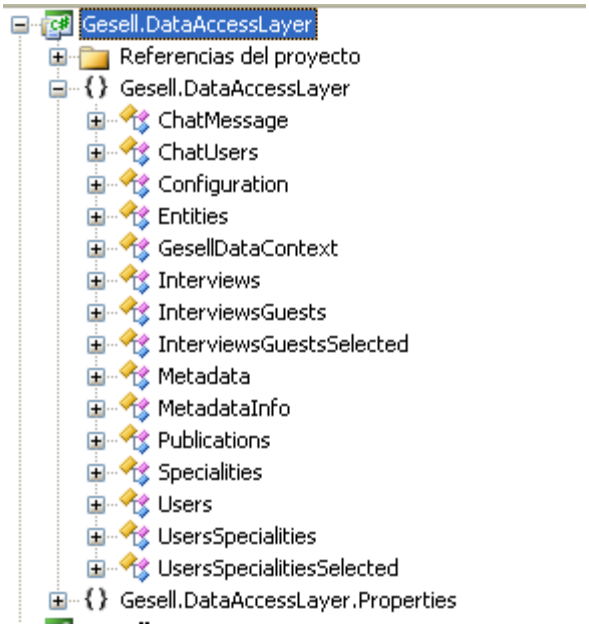


Figura 7: Vista de clases del componente Gesell.DataAccessLayer.

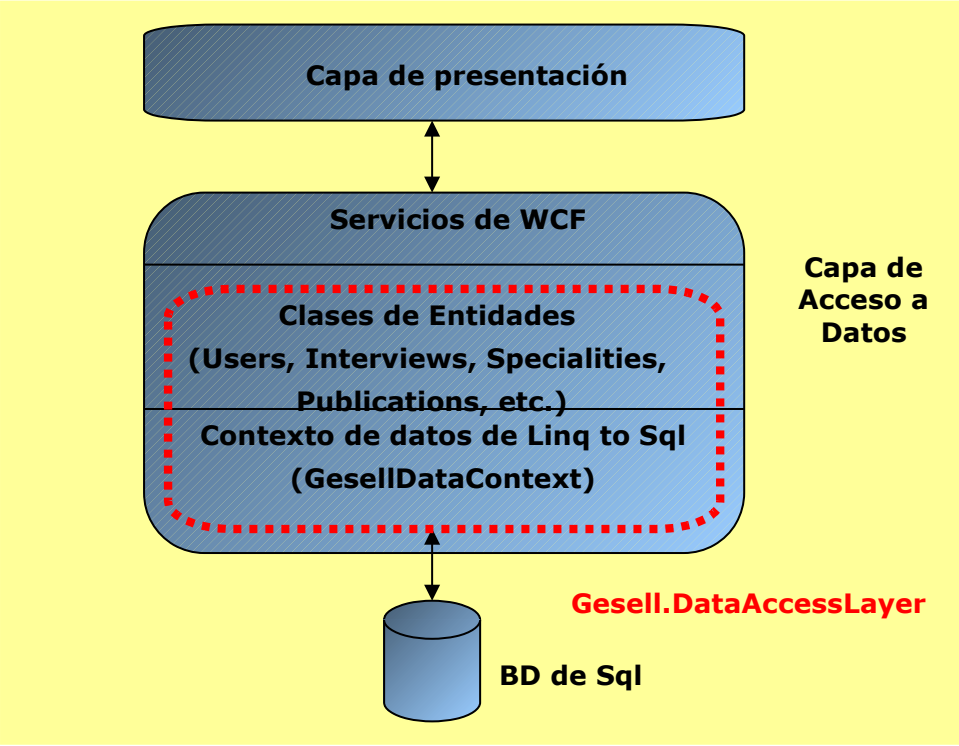


Figura 8: Subcapas de acceso a datos.

6. Operaciones sobre los datos

Para realizar operaciones sobre los datos de la base de datos, las clases entidades utilizan la clase de contexto de datos. Cada instancia de la clase de contexto de datos debería durar una 'unidad de trabajo'. No es recomendable que la instancia se mantenga activa durante mucho tiempo o durante un número arbitrario de transacciones. Por esto, los objetos deben crearse y liberarse en el ámbito de cada método.

6.1 Obtener datos

Para obtener datos desde la base de datos, las clases entidades utilizan la clase de contexto de datos. Para esto se debe instanciar una entidad de esta clase y luego utilizar la sintaxis de consulta de Linq to Sql sobre esta instancia.

En este ejemplo, en la clase entidad *Interviews* se implementa el método Load para retornar un objeto de tipo *Interviews,* asociado al identificador de entrevista *intInterviewId.*

```
//Carga el objeto según la clave.
public Interviews Load(int intInterviewId)
{
  try
  {
    //Contexto de datos.
    GesellDataContext objDB = new GesellDataContext();

    //Obtiene la entrevista.
    IEnumerable<Interviews> objQuery =
objDB.Interviews
              where S.InterviewId.Equals(intInterviewId)
       select S;

    //Retorna el objeto.
    return objQuery.First();
  }
  catch
  {
    throw;
  }
}
```

Código 3: Obtener datos, caso simple.

En un ejemplo un poco más complejo se ve cómo, a través de un mismo contexto de datos, se realiza más de una consulta sobre la base de datos y sobre diferentes entidades. También cómo se retorna más de un objeto y de diferentes tipos. En la misma clase *Interviews* se implementa una sobrecarga del método Load para retornar el mismo objeto de tipo *Interviews* asociado al identificador de entrevista *intInterviewId;* Y una lista de objetos *InterviewsGuestsSelected* asociados a la entrevista.

```
//Retorna entrevista y lista de usuarios invitados asociados, según la clave.
public Interviews Load(int intInterviewId, out List<InterviewsGuestsSelected>
listInterviewsUsersSelected)
{
  try
  {
    IEnumerable<Interviews> objQuery = new List<Interviews>();
     //Se usa en caso de que no exista la entrevista asociada la clave.
    Interviews objTempInterview = new Interviews();
        listInterviewsUsersSelected = new
              List<InterviewsGuestsSelected>();
    //Contexto de datos.
    GesellDataContext objDB = new GesellDataContext();
    if (intInterviewId != 0)
    {
      //Obtiene la entrevista.
      objQuery = from U in objDB.Interviews
            where U.InterviewId == intInterviewId
            select U;
    }
    if (objQuery.Count() > 0)
    {
              //Si la entrevista existe, obtiene los usuarios invitados
                    con la seleccion asociada.
                    IEnumerable<InterviewsGuestsSelected>
                    objQueryInterviewsGuestsSelected =
                    from U in objDB.InterviewsGuestsSelected
                     where U.InterviewId == intInterviewId && U.SpecialityId ==
                     objQuery.First().SpecialityId
                     orderby U.UserDesc
                    select U;
                    //Retorna la lista.
                    listInterviewsUsersSelected =
                    objQueryInterviewsGuestsSelected.ToList();
    }
    if (intInterviewId != 0)
    {
      //Retorna la entrevista.
      return objQuery.First();
    }
    else
    {
      return objTempInterview;
    }
  }
  catch
  {
    throw;
  }
}
```

Código 4: Obtener datos, caso complejo.

6.2 Alta de datos

Para dar de alta registros nuevos, las clases entidades utilizan la clase de contexto de datos junto con el método *InsertOnSubmit.* Luego se ejecutan los cambios con el método *SubmitChanges.*

```
        //Salva el objeto. Se usa para alta.
    public void Save(Users objUser)
    {
      try
      {
          //Contexto de datos.
          GesellDataContext objDB = new GesellDataContext();

          //Se da de alta el usuario.
          objDB.Users.InsertOnSubmit(objUser);
          objDB.SubmitChanges();
      }
      catch
      {
        throw;
     }
        }
```

Código 5: Alta de datos.

En este ejemplo se quiere insertar a la base de datos un objeto de tipo *Users* y varios del tipo *UsersSpecialities*. Linq to Sql comprueba en este caso una transacción implícita. Por lo tanto crea una transacción local para ejecutar todos los comandos necesarios al cambio de objetos planteado. Tanto la inserción del objeto *Users* como la de los objetos *UsersSpecialities* se realiza transaccionalmente.

```
//Salva el objeto y las especialidades asociadas. Se usa para alta.
public void Save(Users objUser, List<UsersSpecialities> listUsersSpecialities)
{
  try
  {
    //Contexto de datos.
    GesellDataContext objDB = new GesellDataContext();

    //Inserta usuario.
    objDB.Users.InsertOnSubmit(objUser);

    //Inserta relacion para especialidades asociadas.
        foreach (UsersSpecialities objUsersSpecialities in listUsersSpecialities)
    {
              objDB.UsersSpecialities.
                    InsertOnSubmit(objUsersSpecialities);
    }
        //Confirma los cambios.
    objDB.SubmitChanges();
  }
  catch
  {
    throw;
  }
}
```

Código 6: Alta de datos transaccional.

6.3 Modificar datos

Para modificar datos, las clases entidades utilizan la clase de contexto de datos. Se debe asociar la entidad a ser modificada a un contexto de datos. Para esto se usa el método Attach.

Cuando una entidad va a ser modificada se realizan comprobaciones de simultaneidad optimista. Para esto existe tres formas de hacerlo, basadas en:

- Marcas de tiempo.
- Valores originales de un subconjunto de propiedades de entidad.

- Entidades completas original y modificada.

Marcas de tiempo

Al modificar una propiedad se actualiza la versión automáticamente, que es usada por la clase de contexto de datos para realizar la comprobación de simultaneidad optimista. En este caso al método Attach recibe la entidad y un parámetro en true para indicar que se usa la marca de tiempo. Luego se confirman los cambios, con el método *SubmitChanges.*

```
objDB.Users.Attach(objNewUser, true);
       objDB.SubmitChanges();
```

Código 7: Modificación de datos, marcas de tiempo.

Valores originales de propiedades de entidad

Se asocia la entidad a ser modificada al contexto de datos y luego se modifican un subconjunto de las propiedades. En este caso el método Attach recibe la entidad y un parámetro en false para indicar que no se usan marcas de tiempo, sino comparación entre el objeto asociado al contexto de datos y los nuevos valores del mismo modificados posteriormente. Luego se confirman los cambios.

```
objDB.Users.Attach(objNewUser, false);
objNewUser.UserName=strUserName;
objNewUser.UserSecondName=strUserSecondName;
objDB.SubmitChanges();
```

Código 8: Modificación de datos, valores originales de propiedades de entidad.

Entidades completas original y modificada

Se asocian al contexto de datos dos objetos, uno que es el original, el obtenido de la base de datos; y otro que es el que tiene las propiedades que han sido modificadas. Luego se confirman los cambios. La clase de contexto de datos compara ambos objetos para realizar la actualización de la base de datos.

Esta opción es la que preferimos usar en el sistema, para el caso de las actualizaciones. La manera en que se usan los objetos de esta clase cuando se quieren actualizar sus datos es:

Se obtiene el objeto invocando al método Load de la clase entidad. Se guarda una copia del mismo para poder ser pasado como parámetro como la entidad original. Se modifican las propiedades del objeto obtenido por el método Load. Este objeto será el que se pase como parámetro como entidad modificada.

```
//Salva el objeto. Se usa para modificación.
public void Save(Users objNewUser,Users objOriginalUser)
{
  try
  {
     //Contexto de datos.
     GesellDataContext objDB = new GesellDataContext();

          //Se pasa entidad modificada y entidad original para  establecer
          los cambios.
     objDB.Users.Attach(objNewUser, objOriginalUser);
     objDB.SubmitChanges();
  }
  catch
  {
    throw;
  }
}
```

Código 9: Modificación de datos, entidades completas original y modificada.

6.4 Eliminar datos

Al igual que en la modificación, para eliminar datos, las clases entidades utilizan la clase de contexto de datos. Se debe asociar la entidad a ser eliminada a un contexto de datos. Se usa el mismo método Attach, pero se indica que el objeto no tiene marca de tiempo. Como no se modifica ninguna propiedad luego de haberlo asociado al contexto de datos, las comparaciones con los valores del registro en la base de datos no generan diferencias.

En el siguiente ejemplo, se implementa el método *Delete* para los objetos de tipo *Metadata*, en donde se ve el uso de lo indicado.

Código 10:

```
//Elimina el objeto.
public void Delete(Metadata objMetadata)
{
  try
  {
    //Contexto de datos.
    GesellDataContext objDB = new GesellDataContext();

    //Elimina.
    objDB.Metadata.Attach(objMetadata, false);
    objDB.Metadata.DeleteOnSubmit(objMetadata);
    objDB.SubmitChanges();
  }
  catch
  {
    throw;
  }
      }
```

Eliminar datos.

7 Sintaxis de Linq to Sql

7.1 Palabras reservadas

Las siguientes son palabras reservadas para realizar las consultas en Linq to Sql.

- From - in: para indicar tabla origen.
- Join – in – on - equals: para indicar join entre tablas.
- Where: para indicar filtro de condición.
- Orderby: para indicar orden.
- Select: para indicar campos a ser seleccionados.
- ExecuteQuery: para ejecutar sentencias sql.

7.2 Ejemplos

Algunos ejemplos de sintaxis usados en las clases de entidades son:

- Seleccionar una entidad

```
IEnumerable<Interviews> objQuery =
       from U in objDB.Interviews
   where U.StatusId == intStatus
   orderby U.InterviewId
          select U;
```

Código 11: Seleccionar entidad.

- Seleccionar una entidad usando join entre tablas

```
IEnumerable<Interviews> objQuery =
       from U in objDB.Interviews
       join I in objDB.InterviewsGuests
   on U.InterviewId equals I.InterviewId
       where U.StatusId == intStatus && U.SpecialityId == intSpecialityId &&
       I.UserId == strUserId
   orderby U.InterviewId
          select U;
```

Código 12: Seleccionar entidad usando join.

- Ejecución de sentencias sql

```
objDB.ExecuteQuery(typeof(string), "delete from MetadataInfo where InterviewId
= " + objInterviews.InterviewId +
          " and FieldId in (select FieldId from Metadata where InterviewId = "
+ objInterviews.InterviewId +
                " and TypeId = " + intTypeId + ")" );
```

Código 13: Ejecución de sentencias sql.

- Funciones

En este caso se ve el uso de máximo y de contar.

```
objQuery = (short)(from U in objDB.Specialities
              select U.SpecialityId).Max();

objQueryInt = (from U in objDB.Metadata
        select U.FieldId).Count();
```

Código 14: Funciones.

8. Referencias

- Aplicaciones remotas y de n niveles con Linq to Sql – último acceso: 30/06/09
  http://msdn.microsoft.com/es-es/library/bb882661.aspx
- Ejemplos de sentencias Linq to Sql – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/vbasic/bb688085.aspx
- Información general (Linq to Sql) – último acceso: 30/06/09
  http://msdn.microsoft.com/es-es/library/bb386974.aspx
- Linq to Sql – último acceso: 30/06/09
  http://msdn.microsoft.com/es-es/library/bb386976.aspx
- Recuperación de datos y operaciones CUD en aplicaciones de niveles N (Linq to Sql) – último acceso: 30/06/09
  http://msdn.microsoft.com/es-es/library/bb546187.aspx

3. Componente Local

La aplicación del proyecto no sólo permite consultar información existente remotamente o ver entrevistas en vivo. También permite realizar descargas locales de todas aquellas entrevistas a las que el usuario fue invitado y que posteriormente podrá ver a demanda.

Para esto es necesario poder realizar la transferencia de la siguiente información:

- Desde el servidor de archivos, transferir un archivo .zip que contiene una imagen de vista previa de la entrevista, un archivo de video y audio en donde se guardo la entrevista y un archivo XML con información sobre los mensajes enviados y recibidos durante la emisión de la entrevista.
- Desde el servidor de base datos, transferir la información asociada a la entrevista, así como información de usuario, especialidad y metadata asociados a la entrevista.

Los archivos incluidos en el archivo .zip serán guardados en una carpeta local con camino relativo /Download/Emisiones, bajo la carpeta donde se encuentra el archivo ejecutable de *G-Sell*.

Los datos asociados a la entrevista serán guardados en una base local embebida en el sistema llamada GesellLocal.sdf. Corresponde a una base de datos Sql Server Compact Edition.

1. Archivos locales

En una carpeta local se descomprimen los archivos incluidos en el archivo .zip que se transfiere desde el servidor de archivos. Los archivos son:

- A.jpg
- A.xml
- A.wmv

El nombre del archivo sigue una nomenclatura especial.

El archivo jpg será usado para la vista previa de la entrevista, mientras que los archivos xml y wmv contienen la grabación de la entrevista. Ver Anexo – Formato de archivos multimedia.

2. Sql Server Compact Edition

Microsoft Sql Server 2005 Compact Edition es un motor de base de datos ligero que puede ser usado en aplicaciones de escritorio. Las operaciones sobre los datos se realizan a través de sintaxis Sql similar a Sql Server; usando una API que expone servicios análogos a una versión Sql Server instalada en un servidor de base de datos.

Para trabajar con bases de datos de Sql Server Compact Edition, se accede a ellas desde el IDE de desarrollo de Visual Studio, a través del Explorador de Servidores. Allí se crea una conexión indicando que el proveedor es Microsoft Sql Server 2005 Compact Edition y se indica el directorio local en el cual se creará la base, que tendrá extensión sdf. Luego es posible desde el mismo explorador, crear tablas y vistas; acceder a los datos y realizar las operaciones necesarias.

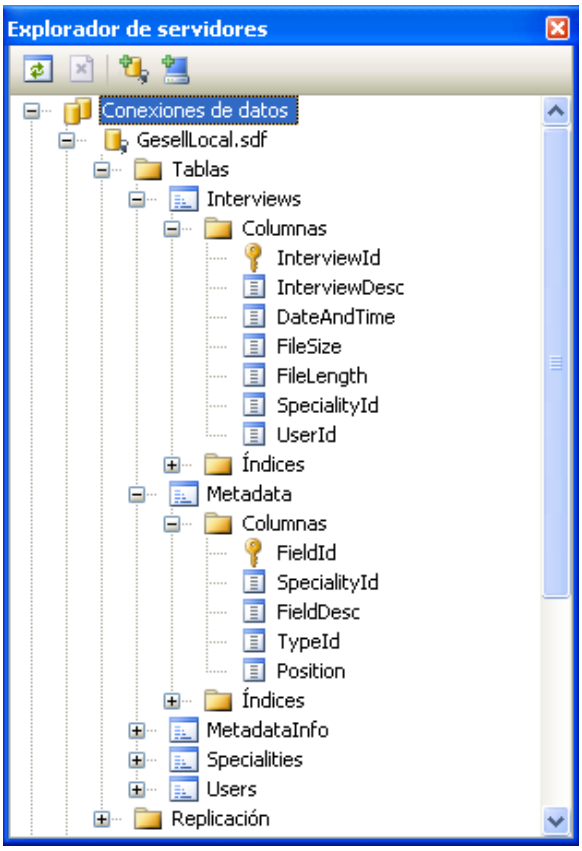


Figura 1: Base de datos local en explorador de servidores.

2.1 Sintaxis

La sintaxis que se usa para acceder a los datos de una base Sql Server Compact Edition es Sql. Algunos ejemplos de operaciones sobre los datos de la base son:

- Insertar en una tabla.

```
//Métodos del mantenimiento.
public void Save(SqlCeConnection objConnection)
{
  //Salva usuario dentro de una transaccion.
  try
  {
    string strSql = "insert into Specialities (SpecialityId,
              SpecialityDesc)" +
      " values (" + mintSpecialityId + ", '" + mstrSpecialityDesc + "'
              )";
    SqlCeCommand objCommand = new SqlCeCommand(strSql, objConnection);
    objCommand.ExecuteNonQuery();
  }
  catch
  {
    throw;
  }
      }
```

Código 1: Insertar registro.

- Eliminar un registro de una tabla.

```
public void Delete(SqlCeConnection objConnection, int intInterviewId)
{
  //Elimina la entrevista.
  try
  {
    string strSql = "delete from Interviews where InterviewId = " +
              intInterviewId.ToString();
    SqlCeCommand objCommand = new SqlCeCommand(strSql, objConnection);
    objCommand.ExecuteNonQuery();
  }
  catch
  {
    throw;
  }
      }
```

Código 2: Eliminar registro.

- Obtener un registro de una tabla y uso de los valores obtenidos.

```
SqlCeConnection objConnection = new SqlCeConnection(strConnection);
objConnection.Open();
string strSql = "select * from Interviews where UserId = '" + strUserId + "'";
SqlCeCommand objCommand = new SqlCeCommand(strSql, objConnection);
SqlCeDataReader objResult = objCommand.ExecuteReader();

List<Interviews> listInterviews=new List<Interviews>() ;

try
  {
    while (objResult.Read())
      {
        Interviews objInterviews = new Interviews();
        objInterviews.InterviewId= objResult.GetInt32(0);
        objInterviews.InterviewDesc = objResult.GetString(1);
        objInterviews.DateAndTime= objResult.GetDateTime(2);
```

Código 3: Obtener registro.

3. Base de datos local

La base local de Sql Server Compact Edition que mantiene los datos del sistema se llama GesellLocal.sdf. Básicamente contiene un subconjunto de información de la base remota Gesell de Sql Server 2005 Express.

3.1 Estructura

Algunas de las tablas de la base GesellLocal.sdf tienen la misma estructura que la base *Gesell*. Otras están compuestas por alguno de los campos y no por todos. Además se cuenta con una tabla que sólo se usa de manera local. Otro grupo de tablas de *Gesell* no son incluidas en la base local. Se detalla la estructura de cada tabla incluida de la base local y las diferencias con la base remota.

Users

	Nombre	Tipo	Descripción
🔑	UserId	Alfanumérico (20)	Identificador de usuario.
	UserName	Alfanumérico (40)	Nombre(s) del usuario.
	UserSecondName	Alfanumérico (40)	Apellido(s) del usuario.

El resto de los campos de esta tabla incluidos en la base remota no son tenidos en cuenta para preservar la seguridad y porque no es información relevante para ser asociada a los datos de la entrevista que se descarga. Si es necesaria información descriptiva del usuario.

Specialities

	Nombre	Tipo	Descripción
🔑	SpecialityId	Numérico Entero	Identificador de especialidad.
	SpecialityDesc	Alfanumérico (40)	Descripción de especialidad.

En este caso, la estructura de la tabla local y remota son las mismas.

Interviews

	Nombre	Tipo	Descripción
🔑	InterviewId	Numérico Entero	Identificador de la entrevista.
	InterviewDesc	Alfanumérico (40)	Descripción de la entrevista.
	DateAndTime	Fecha Hora	Fecha y hora de realizada la entrevista.
	FileSize	Numérico Entero	Tamaño del archivo expresado en kbytes.
🔑	SpecialityId	Numérico Entero	Identificador de especialidad.
🔑	UserId	Alfanumérico (20)	Identificador de usuario.

Tanto el campo StatusId como el campo StatusEntity existentes en la base remota y que indican el estado de la entrevista no se incluyen en la tabla local. Esto es porque el estado de todas las entrevistas que se pueden descargar es 'Finalizada'. De esta manera se evita información redundante e implícita.

El resto de los campos de esta tabla no son tenidos en cuenta por no ser información relevante para ser asociada a los datos de la entrevista que se descarga.

MetadataInfo

	Nombre	Tipo	Descripción
🔑	InterviewId	Numérico Entero	Identificador de la entrevista.
🔑	FieldId	Numérico Entero	Identificador del campo.
	Data	Alfanumérico (1000)	Información.

La información asociada a cada entrevista depende de la definición de metadata que se haga para la especialidad asociada a la entrevista. En esta tabla se mantiene la información referente a los campos configurables definidos en la metadata de la especialidad. A diferencia de los campos incluidos en la tabla de entrevistas (Interviews), la información que se mantiene en esta tabla, es configurable a través del sistema. Los campos de Interviews, son fijos para todas las entrevistas. Igualmente, los datos aquí definidos junto con los datos definidos en la tabla Interviews, conforman toda la información respecto de una entrevista.

En este caso, la estructura de la tabla local es igual a la de la base remota.

Metadata

	Nombre	Tipo	Descripción
🔑	FieldId	Numérico Entero	Identificador de campo.
🔑	SpecialityId	Numérico Entero	Identificador de especialidad.

	FieldDesc	Alfanumérico (40)	Descripción del campo visible en la interfaz de usuario. (*2)
	TypeId	Numérico Entero	Identificador de tipo de campo. (*1)
	Position	Numérico Entero	Posición del campo, establece el orden en que se visualizan los campos.

(*1) Permite indicar si un campo se usa en la portada de la entrevista ó en el cierre de la misma. Los valores posibles son:

1- Portada

2- Cierre

En la base de datos remota se tiene asociado a este campo otro llamado TypeEntity. Como los campos de tipo Entidad se usan para poder resolver cuestiones de Linq to Sql, no son necesarias en la base local.

(*2) Asociado a este campo se tiene en la base de datos del servidor otro llamado MaxLength que permite indicar el largo máximo del campo. Como el uso local de la Metadata sólo es descriptivo, no se permiten modificaciones de la entrevista, este campo no se tiene en cuenta.

No se incluye el campo llamado InUse de la base remota, que permite indicar si el campo está en uso todavía. Esto se usa en la edición de una entrevista. Por tal motivo no se tiene en cuenta en la base local donde la información no es editable por el usuario. Los campos en uso serán todos los incluidos en la metadata de la entrevista. Los campos a los cuales no se les haya asignado valor para la entrevista ni siquiera tienen registros en la base remota.

InterviewsUsersDownload

	Nombre	Tipo	Descripción
	InterviewId	Numérico Entero	Identificador de la entrevista.
	UserId	Alfanumérico (20)	Identificador del usuario que descarga la entrevista.

Permite indicar que usuarios han descargado cual entrevista en forma local en un equipo cliente. De esta manera, si más de un usuario tuviera acceso a un mismo equipo, tendrá asociadas las entrevistas que el descargo, y no la de otros usuarios.

Esta tabla no existe en la base de datos del servidor. Solamente se usa de forma local.

4. Acceso a datos

Para el acceso a datos locales del sistema se crea un nuevo componente dentro de la arquitectura del sistema. Este componente compone la capa de acceso a datos. Su nombre es Gesell.DataAccessLayerLocal.

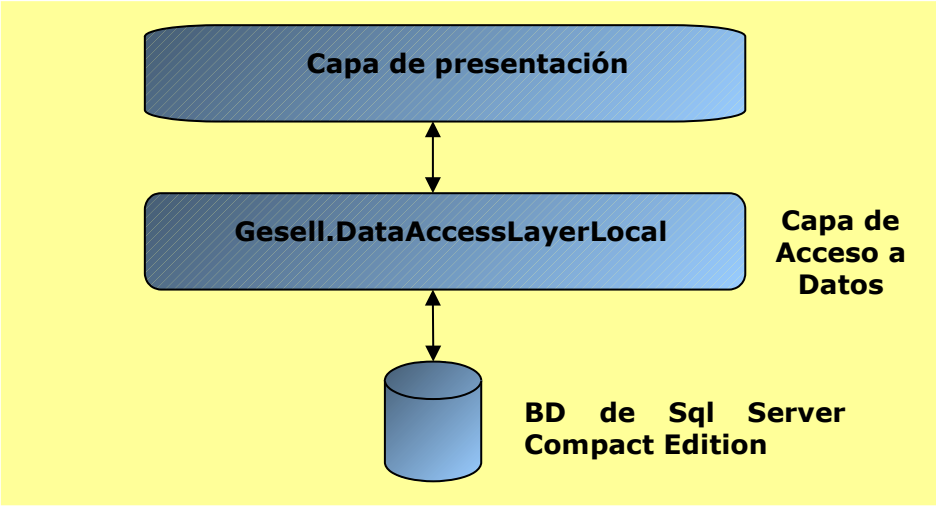


Figura 2: Capa de acceso a datos.

4.1 Componente

Gesell.DataAccessLayerLocal contiene una clase asociada a cada tabla de la base: Interviews, Metadata, MetadataInfo, Specialities y Users. Estas clases mantienen propiedades asociadas a los campos de las tablas y tienen los métodos necesarios para actualizar y obtener información de cada una de las tablas. El componente cuenta además con una clase llamada TotalInterview que centraliza la lógica de la actualización de la base de datos local cuando ocurren descargas de entrevistas. Además se cuenta con la clase InterviewsUsersDownload para tener información de las descargas realizadas localmente por cada usuario.

En el siguiente diagrama de clases puede visualizarse las clases mencionadas, con las variables locales de cada clase, y las propiedades y métodos que expone.

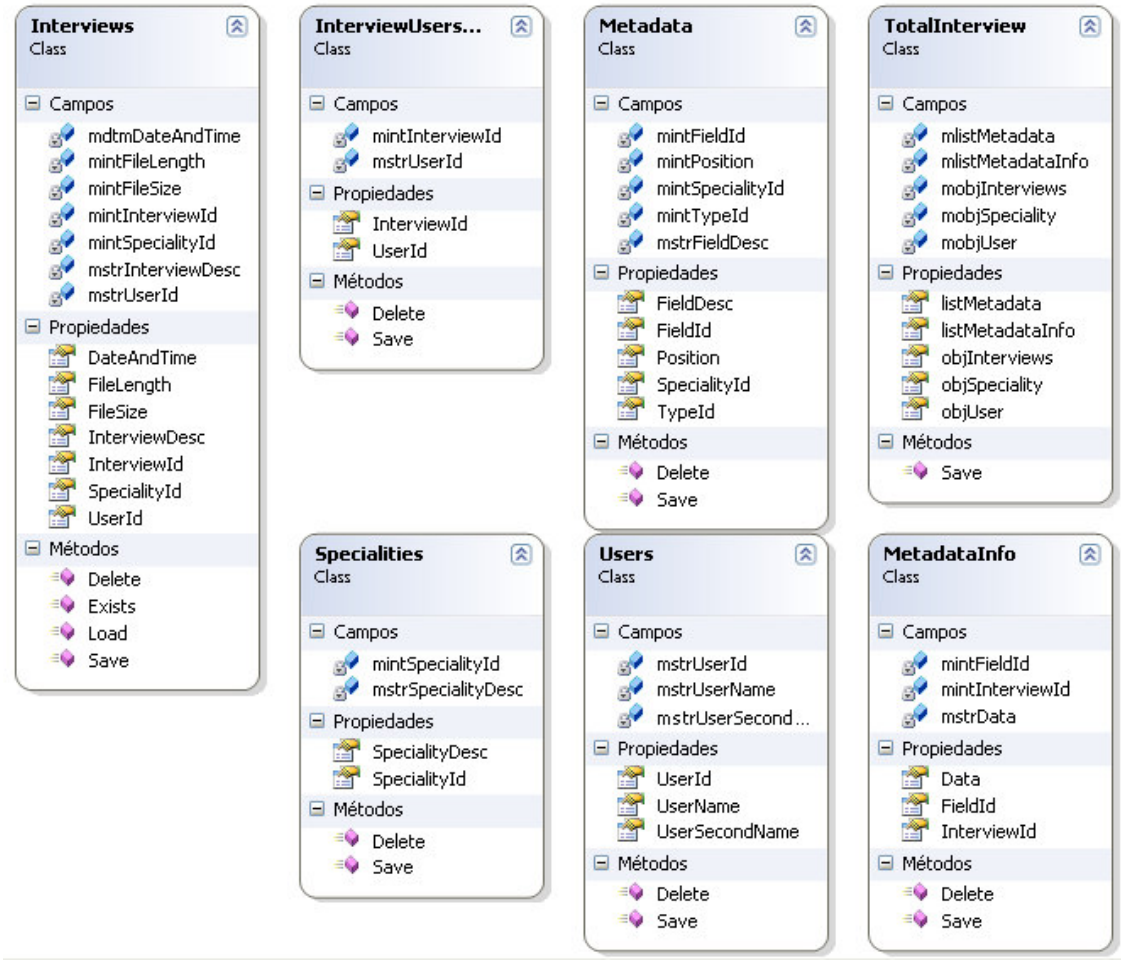


Figura 3: Diagrama de clases.

4.2 Transferencia

Se transfiere información de la base *Gesell* remota a la base local GesellLocal.sdf cada vez que se realiza una descarga de una entrevista. La lógica utilizada se puede ver en el siguiente pseudo-código.

1- Se obtiene información de la entrevista que se esta descargando desde la base de datos remota.
2- Se establece la conexión a la base de datos local.
3- Se comienza una transacción.
4- Inserta/Actualiza la especialidad de la entrevista que se esta descargando.
5- Inserta/Actualiza el usuario que publicó la entrevista.
6- Inserta/Actualiza la metadata de la especialidad de la entrevista.
7- Inserta/Actualiza la información de metadata de la entrevista.
8- Inserta/Actualiza la información de la entrevista.
9- Se finaliza la transacción.
10- Se cierra la conexión a la base de datos local.

1- Se obtiene información desde la base de datos remota a través de los servicios web publicados en el servidor de aplicaciones.

2- La base de datos local está ubicada en una carpeta llamada Base situada en la misma carpeta donde se encuentra el archivo ejecutable de *G-Sell*.

3- Todas las altas o modificaciones realizadas a los datos para la entrevista que se esta descargando se ejecutan transaccionalmente.

4- En caso de actualización, si la descripción de la especialidad hubiera sido modificada en la base remota, estaría desactualizada para todas las entrevistas guardadas localmente. Se actualiza en este momento para todas las entrevistas locales.

5- De igual manera que para las especialidades, se actualiza nombres y apellidos del usuario que publicó la entrevista.

6- En caso de actualización, dado que los registros de metadata no pueden ser eliminados en la base remota, sólo puede darse el caso de tener nuevos registros para la metadata de la especialidad de la entrevista; o que la descripción de alguno de ellos hubiera variado.

7 y 8- Una vez que una entrevista es publicada y queda disponible para ser descargada no puede modificarse su información; ni la fija, ni la asociada a la metadata. Entonces no existe la modificación de estos datos, en cuanto a sus valores se refiere. Lo que se hace es eliminar los registros de la base local asociados a esta información para volver a insertarla. Se evita de esta manera errores por registros duplicados y se soluciona cualquier inconveniente que hubiera existido en una descarga anterior de la misma entrevista.

Se permite descargar una entrevista más de una vez. En este caso luego de consultar al usuario, se permite modificar la información de la entrevista localmente. Esto puede ser requerido por el usuario en caso de que, por ejemplo, el archivo de video no funcione adecuadamente o este corrupto. Es en estos casos donde los puntos 7 y 8 estarían sobrescribiendo información. Pero se hace de esta manera ya que se puede insertar la información en la base de datos en forma transaccional, pero no se puede agregar a este procedimiento la descarga de los archivos requeridos.

5. Referencias

- Microsoft Sql Server Compact Edition Linq to Sql – último acceso: 30/06/09
http://www.microsoft.com/Sqlserver/2005/en/us/compact.aspx

A4 – Anexos referidos a video y streaming

1. Captura y streaming multimedia

1. Introducción

La componente de captura y streaming de contenido multimedia que representa la emisión de la entrevista se diseño e implementó basada en las funciones que brindan las librerías DirectShow y WindowsMediaFormat de Microsoft.

Para adecuar el uso de estas librerías al leguaje seleccionado (C sharp) se debió recurrir a proyectos de distribución libre DirectShow.NET y WindowsMediaLib.NET que implementan las versiones en .NET correspondientes.

Debido a incapacidad de WPF estándar de manejar grafos DirectShow personalizados como es el caso de la captura de audio y video se debió recurrir al trabajo en constante desarrollo de Jeremiah Morril denominado MediaKit. La librería MediaKit permite construir fácilmente controles para reproductores de multimedia en WPF. El kit incluye, entre otros, un reemplazo para el control MediaElement y un VideoCaptureElement para capturas de video desde cámaras web. Aprovechando la base desarrollada en esta librería se construyó un control que permitiera además, la captura de audio y el envió del contenido multimedia capturado a un servidor de streaming. Se mantuvo la estructura ya existente, usando en el código modificado comentarios en idioma inglés previendo futuros interesados en mejorar el presente trabajo.

2. Construcción del grafo

La emisión de una entrevista requiere que el flujo multimedia capturado sea grabado localmente y enviado en forma simultanea hacia el servidor de streaming. Para ello se debe construir un grafo DirectShow que permita realizar esta compleja labor.

Dada la dinámica de una entrevista y los eventos que suceden en ella, se dividió la tarea en dos fases.

Una etapa previa a la transmisión donde el emisor se prepara, selecciona el dispositivo audio, el de video, ajusta volumen, prueba la captura, entre otras verificando las condiciones para que la emisión en vivo sea óptima.

Luego la fase de emisión propiamente dicha que es cuando la entrevista se desarrolla en vivo. Aquí el grafo ya debe incluir filtros apropiados para el envío del flujo capturado y mecanismos de autenticación contra el servidor WMS debido a las restricciones de seguridad impuestas.

2.1 Construcción del grafo de captura

La etapa de construcción del grafo de captura consta de inicializar el grafo e ir definiendo y agregando filtros para captura de audio y video que una vez conectados adecuadamente forman el elemento de captura multimedia.

```
/* Create a new graph */
m_graph = (IGraphBuilder)new FilterGraph();

/* Create a capture graph builder to help
 * with rendering a capture graph */
var captureGraph =(ICaptureGraphBuilder2) new CaptureGraphBuilder2();

/* Set our filter graph to the capture graph */
int hr = captureGraph.SetFiltergraph(m_graph);
DsError.ThrowExceptionForHR(hr);
```

Código 1: Inizialización del grafo.

Con la instancia del grafo definida, se puede comenzar la incorporación de filtros source que permitan la captura y filtros de entrega que permitan la visualización del video capturado.

```
/* Add our capture device source to the graph */
m_captureDevice = AddFilterByName(m_graph,
                                  FilterCategory.VideoInputDevice,
                                  VideoCaptureSource);

/* Add our audio capture device source to the graph */
m_captureAudioDevice = AddFilterByName(m_graph,
                                       FilterCategory.AudioInputDevice,
                                       AudioCaptureSource);

/* Creates a video renderer and register
 * the allocator with the base class */
m_renderer =CreateVideoRenderer(VideoRendererType.VideoMixingRenderer9)

/* Add the renderer */
m_graph.AddFilter(m_renderer, "VMR9 Renderer");
```

Código 2: Agrega filtros de captura y entrega.

Ya pensando a futuro y a los efectos de duplicar las capturas, se insertan en el grafo filtros de transformación especiales cuya función es generar tantas salidas como se necesiten a partir del flujo de entrada que reciben.

```
/* Add the Video infinite Tee fliter */
infTeeVideo = (IBaseFilter)new InfTee();
hr = m_graph.AddFilter(infTeeVideo, "Infinite Tee Video");
DsError.ThrowExceptionForHR(hr);

/* Add Audio infinite Tee fliter */
infTeeAudio = (IBaseFilter)new InfTee();
hr = m_graph.AddFilter(infTeeAudio, "Infinite Tee Audio");
DsError.ThrowExceptionForHR(hr);
```

Código 3: Agrega filtros de copia de flujo.

Por último, para completar la construcción del grafo de captura es necesario conectar estos filtros en el orden apropiado para que luego el flujo de datos circule de un extremo a otro. Para realizar esta tarea se usan los dos métodos de conexión disponibles en DirectShow. Los filtros de captura de audio y video a ambos filtros de copia se conectan en forma manual por medio de pines de entrada y salida.

```
/* Connect Video output pin to
 * Infinite tee fliter input */
IPin outPin = DsFindPin.ByCategory(m_captureDevice,
                                   PinCategory.Capture, 0);
IPin inPin = DsFindPin.ByDirection(infTeeVideo, PinDirection.Input, 0);
hr = m_graph.Connect(outPin, inPin);
DsError.ThrowExceptionForHR(hr);

/* Connect Audio output pin to
 * Infinite tee fliter input */
outPin= DsFindPin.ByCategory(m_captureAudioDevice,
                             PinCategory.Capture,0);
inPin = DsFindPin.ByDirection(infTeeAudio, PinDirection.Input, 0);
hr = m_graph.Connect(outPin, inPin);
DsError.ThrowExceptionForHR(hr);
```

Código 4: Conexión manual de pines de entrada y salida.

Los enteros definidos en las direcciones de los pines ya sea de entrada o salida (en este caso 0), corresponden al número de orden de cada pin en el filtro.

Mientras que las salidas de los filtros de copia son conectados a los filtros que representan la entrega del flujo al usuario final en forma automática.

```
/* Intelligent connection to Video Render */
hr = captureGraph.RenderStream(null, MediaType.Video,
                               infTeeVideo, null, m_renderer);
DsError.ThrowExceptionForHR(hr);


/* Intelligent connection to Audio Render */
hr = captureGraph.RenderStream(null, MediaType.Audio,
                               infTeeAudio, null, null);
DsError.ThrowExceptionForHR(hr);
```

Código 5: Conexión automática de filtros.

Al terminar el proceso de armado se obtiene un grafo DirectShow de captura multimedia como el que se muestra en la siguiente figura.

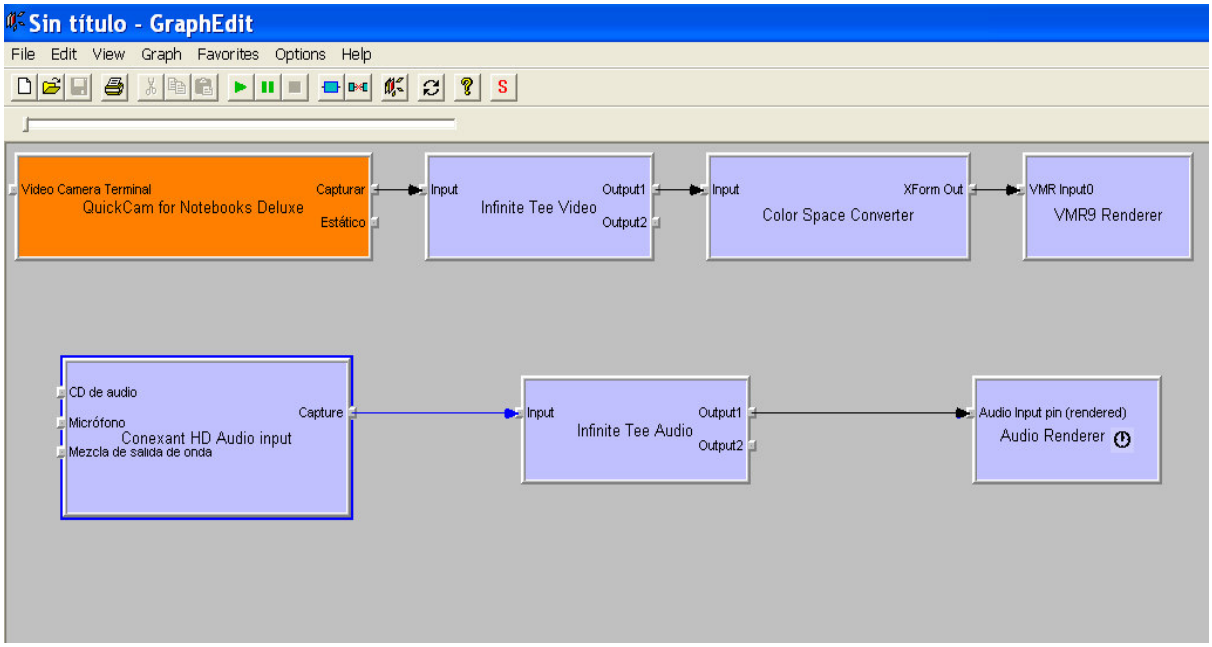


Figura 1: Grafo DirectShow de captura.

2.2 Construcción del grafo de emisión

En el momento que el usuario emisor considera que tiene el entorno preparado y los dispositivos ajustados para comenzar la entrevista, este inicia la emisión en vivo.

A nivel de DirectShow, esto implica agregar al grafo ya construido un par de filtros que permitan la grabación y emisión del flujo de datos al servidor de streaming WMS.

En este punto, se debe hacer uso de la librería WindowsMediaLib (Windows Media Format SDK para .NET) y la funcionalidad que esta brinda para el tratamiento de flujos multimedia.

Se deben configurar los distintos parámetros de los filtros antes mencionados para obtener el formato elegido. Existe más de una forma de configurarlos, pero todas terminan en un archivo tipo PRX que define cada parámetro y su valor correspondiente, formando así, perfiles que permiten dar formato a un flujo determinado. Estos perfiles pueden ser cargados al filtro en cuestión directamente por medio de su GUID (identificador único en el sistema), o a través un método que lo configura desde un archivo especialmente preparado. Luego de varias pruebas realizadas de optó por configurar el formato de salida con el GUID definido en forma estándar por Windows Media en el archivo WMSysPr9.prx.

```
              .
              .
              .
<profile version="524288"
       name="Windows Media Video 8 for Broadband (NTSC, 1400 Kbps)"
       guid="{931D1BEE-617A-4bcd-9905-CCD0786683EE}"
       description="Use this profile for high quality audio/video content
intended for local playback or for download and playback.">
  <streamconfig majortype="{73647561-0000-0010-8000-00AA00389B71}"
         streamnumber="1"
         streamname="Audio Stream"
         inputname="Audio"
         bitrate="128040"
         bufferwindow="-1">
              .
.
   <streamconfig majortype="{73646976-0000-0010-8000-00AA00389B71}"
          streamnumber="2"
          streamname="Video Stream"
          inputname="Video"
          bitrate="1272000"
          bufferwindow="5000">
    <videomediaprops maxkeyframespacing="100000000"
            quality="75"/>
    <wmmediatype subtype="{32564D57-0000-0010-8000-00AA00389B71}"
          bfixedsizesamples="0"
          btemporalcompression="1"
          lsamplesize="0">
      <videoinfoheader dwbitrate="1272000"
            dwbiterrorrate="0"
            avgtimeperframe="333333">
              .
.
       <rctarget left="0"
         top="0"
         right="320"
         bottom="240"/>
      <bitmapinfoheader biwidth="320"
               biheight="240"
               biplanes="1"
               bibitcount="24"
               bicompression="WMV2"
                    .
                    ./>
    </videoinfoheader>
   </wmmediatype>
   </streamconfig>
  </profile>
```

Figura 2: Sección del archivo WMSysPr9.prx.

Filtro para grabación

Para poder grabar un archivo a disco conteniendo el flujo capturado por los dispositivos, primero se debe configurar el formato de salida que tendrá el filtro.

En este paso, se configura formato y nombre del archivo obteniendo el filtro resultante y un sink asociado al flujo de grabación.

```
int hr = capGraph.SetOutputFileName(MediaSubType.Asf,
                                    szOutputFileName, out asfWriter,
                                    out pTmpSink);
DsError.ThrowExceptionForHR(hr);

IConfigAsfWriter lConfig = asfWriter as IConfigAsfWriter;
Guid cat = new Guid("{931D1BEE-617A-4bcd-9905-CCD0786683EE}");

hr = lConfig.ConfigureFilterUsingProfileGuid(cat);
       …
return asfWriter;
```

Código 6: Configuración de filtros en formato ASF.

En el anterior código se puede apreciar una vista parcial de la implementación del código que permite configurar el filtro de grabación en formato ASF y agregarlo al grafo ya construido. A pesar de ser el usado, el GUID mostrado aquí es agregado en forma ilustrativa ya que en realidad es pasado como parámetro al método.

Luego de agregado el filtro solo resta conectarlo en forma adecuada para que la información fluya cuando sea necesario.

```
/* Connecting Video to ASF filter*/
outPin = DsFindPin.ByDirection(infTeeVideo,
                               PinDirection.Output, 1);
inPin = DsFindPin.ByDirection(confAsf, PinDirection.Input, 1);
hr = graph.Connect(outPin, inPin);
DsError.ThrowExceptionForHR(hr);

/* Connecting Audio to ASF filter */
outPin = DsFindPin.ByDirection(infTeeAudio,
                               PinDirection.Output, 1);
inPin = DsFindPin.ByDirection(confAsf, PinDirection.Input, 0);
hr = graph.Connect(outPin, inPin);
DsError.ThrowExceptionForHR(hr);
```

Código 7: Conexión de filtro para grabación.

Filtro para emisión

El caso de la emisión en vivo para la captura multimedia es similar al de grabación local, con la salvedad que se necesita definir un perfil de menor calidad que configure el filtro. Siendo así por la restricción ya mencionada de ancho de banda y transmisión sobre TCP del flujo codificado. De esta manera, la configuración, inserción y conexión del filtro ASF en el grafo DirectShow es análoga.

```
/* Configuring Network ASF filter */
confAsfNet = ConfigAsf(captureGraph, "Emisiones/opmet",
                    "D8722C69-2419-4b36-B4E0-6E17B60564E5");

/* Connecting Video to Network ASF filter */
outPin = DsFindPin.ByDirection(infTeeVideo,
                               PinDirection.Output, 2);
inPin = DsFindPin.ByDirection(confAsfNet,
                               PinDirection.Input, 0);
hr = graph.Connect(outPin, inPin);


/* Connecting Audio to Network ASF filter */
outPin = DsFindPin.ByDirection(infTeeAudio,
                               PinDirection.Output, 2);
inPin = DsFindPin.ByDirection(confAsfNet,
                               PinDirection.Input, 1);
hr = graph.Connect(outPin, inPin);
```

Código 8: Definición y conexión de filtro para emisión.

Una vez agregado al grafo, es necesario remover el sink de grabación asociado al filtro y agregar un PushSink que será el responsable de enviar el flujo capturado a través de la red. Este objeto WriterPushSink es configurado con los parámetros necesarios que permiten crear y escribir el punto de publicación en el servidor WMS.

```
// Remove local save Sink
RemoveAllSinks(m_pWriterAdvanced, 1);

// Declare live stream
m_pWriterAdvanced.SetLiveSource(true);

// Create the Network Sink
WMUtils.WMCreateWriterPushSink(out m_pPushSink);

// Add Sink to the IWMWriterAdvanced
m_pWriterAdvanced.AddSink(m_pPushSink as IWMWriterPushSink);
                …
// When Network sink is ready, we open them
m_pPushSink.Connect(Url, null, true);
```

Código 9: Definición de objeto PushSink para emisión.

El string URL está formado por el protocolo, puerto, dirección del servidor de streaming y punto de publicación donde se dejarán el flujo multimedia.

El punto de publicación es creado con las propiedades por defecto seteadas en el servidor de streaming, pero para que el objeto PushSink pueda crearlo es necesario realizar una autenticación contra el servidor WMS.

Autenticación contra servidor WMS

Con el fin de evitar emisiones no deseadas, en el servidor de streaming WMS se ha definido un usuario de sistema llamado General quien exclusivamente podrá crear puntos y emitir una determinada señal.

Para que la aplicación pueda enviar sus credenciales y ser notificada de eventos registrados durante el streaming es necesaria una instancia de comunicación con el WMS. Esto es logrado mediante la implementación de los miembros onStatus de la interfaz IWMStatusCallback y AcquireCredentials de la interfaz IWMCredentialCallBack.

Ambas implementaciones son reunidas en una clase definida con el nombre Autentica que implementa las interfaces antes mencionadas y es agregada al paquete MediaKit.

```
#region IWMCredentialCallback Members
public void AcquireCredentials(string pwszRealm, string pwszSite,
                               char[] pwszUser, int cchUser,
                               char[] pwszPassword, int cchPassword,
                               int hrStatus,
                               ref CredentialFlags pdwFlags)
{

       const string user = "general\0";
   const string pw = "gesell09\0";

   int t = Array.IndexOf(pwszUser, '\0');
   string u = new string(pwszUser, 0, t);
   t = Array.IndexOf(pwszPassword, '\0');
   string p = new string(pwszPassword, 0, t);
   char[] uc = user.ToCharArray();

       // MUST use copy! Don't just assign into pwszUser
   Array.Copy(uc, pwszUser, uc.Length);
   char[] pc = pw.ToCharArray();
   Array.Copy(pc, pwszPassword, pc.Length);
   pdwFlags = CredentialFlags.ClearText;
}
#endregion
```

Código 10: Implementación de AcquireCredentials en clase Autentica.

Es a través de la clase Autentica que la aplicación emisora es habilitada a recibir notificaciones desde el servidor. Para esto es necesario invocar al método Advice de la interfaz IWMRegisterCallback definida sobre el PushSink correspondiente al filtro de de emisión.

```
// Prepare App for authentication and events
Autentica auth = new Autentica();
IWMRegisterCallback pRegCallback = m_pPushSink as IWMRegisterCallback;
pRegCallback.Advise(auth, IntPtr.Zero);
```

Código 11: Preparación de comunicación con WMS.

Una vez completada esta etapa, el grafo DirectShow construido está listo para emitir la señal producida. En la siguiente figura se pude apreciar un esquema del grafo de emisión resultante.

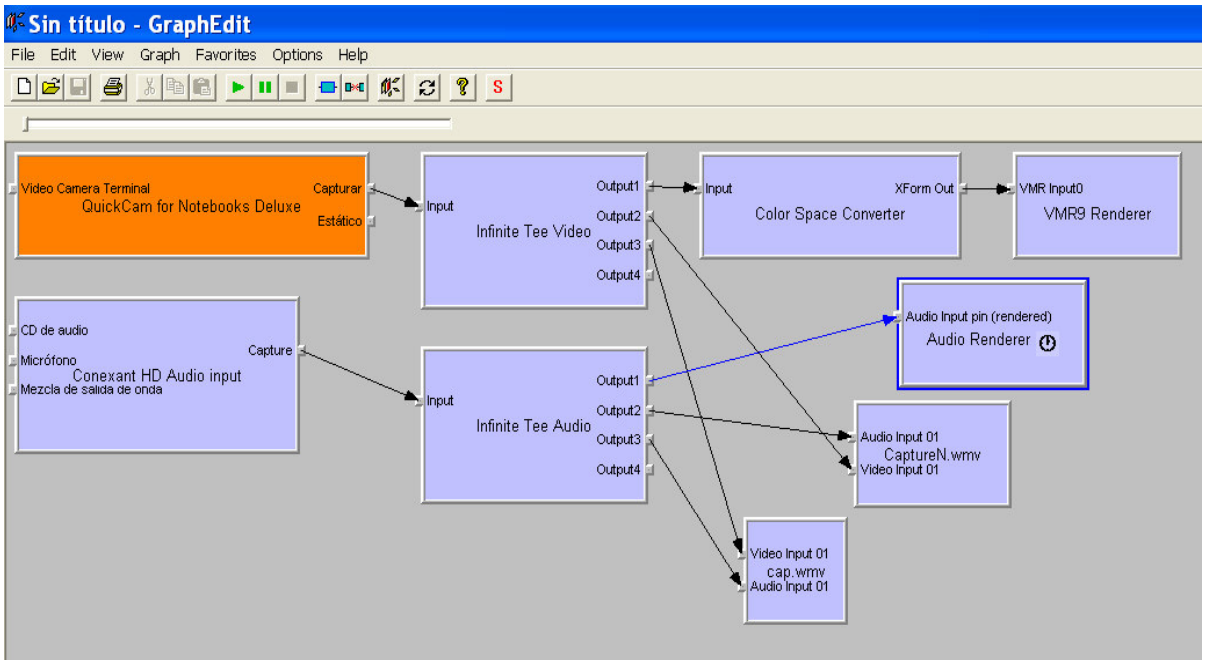


Figura 3: Grafo DirectShow de emisión.

3. Presentación de captura en WPF

Windows Presentation Foundation cuenta en forma estándar con un control para reproductor de contenido multimedia (MediaElement) que es apoyado por el producto Windows Media Player. Dicho control es el que brinda soporte al tratamiento de audio y video permitiendo aplicar las facilidades existentes dentro del universo WPF. Desafortunadamente esta API está dedicada a la reproducción, no permitiendo personalizaciones de ningún tipo, por lo que no es viable en la representación de grafos DirectShow construidos para captura multimedia.

Para solucionar este problema, se recurrió a una solución que aprovecha las nuevas funcionalidades introducidas por Direct3D para la presentación y transformación de elementos en WPF. La librería MediaKit proporciona un conjunto de controles personalizados WPF para el tratamiento de multimedia, que permite incorporar la posibilidad de trabajar con grafos especiales. Dentro de estos controles se puede destacar el VideoCaptureElement que arma y muestra un grafo simple de captura de video desde una cámara web.

Debido a que la emisión de una entrevista involucra realizar más acciones respecto al tratamiento multimedia, como la captura de audio, grabación y transmisión vía red, la complejidad del grafo DirectShow a formar aumenta. Con el fin de modelar una solución al problema planteado se resolvió incorporar un nuevo control a la librería que permita agregar todas estas funcionalidades a la captura ya disponible.

3.1 Modificación a librería MediaKit

En la medida que el VideoCaptureElement no cumplía con todas las funcionalidades necesarias, se utilizó como clase base para crear el CaptureElement. Este nuevo control personalizado agrega nuevos miembros, propiedades y métodos para modelar el comportamiento requerido.

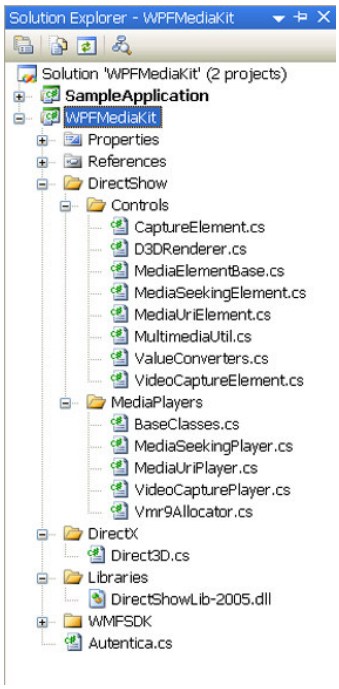


Figura 4: Vista de clases de MediaKit modificada.

Adicionalmente, se debió modificar el compartimiento del VideoCapturePlayer para adaptar el reproductor y poder representar en pantalla el resultado del grafo de emisión construido para *G-Sell*.

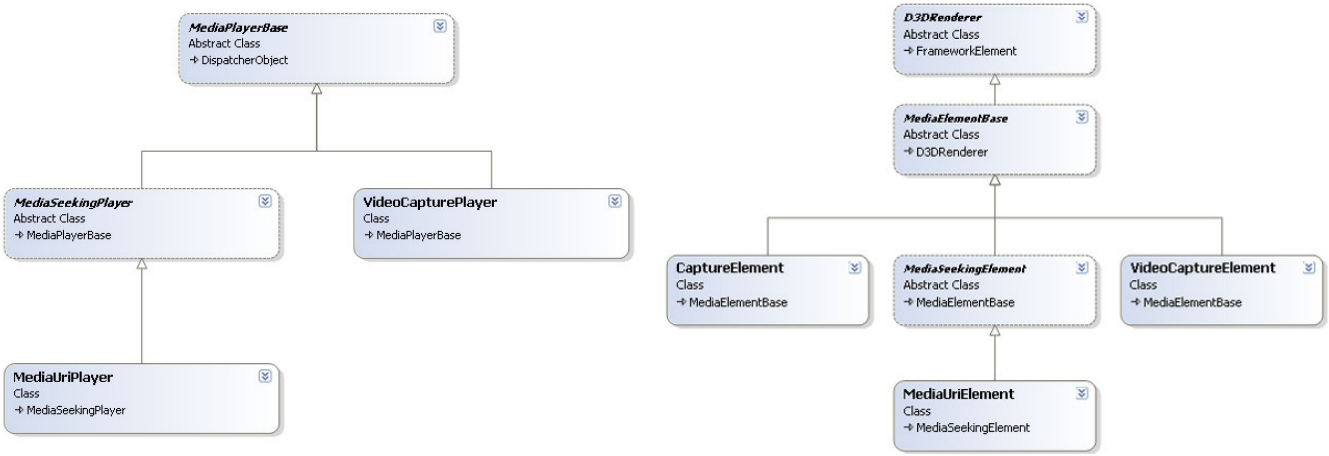


Figura 5: Vista parcial del diagrama de clases.

CaptureElement

Este control se encuentra implementado en la clase CaptureElement.cs donde se debieron agregar miembros y métodos que reflejaran el cambio de estado entre captura inicial y emisión de la entrevista. Para esto se incluyen PlaytoStream, DecideChageGraph y CloseStream entre otros.

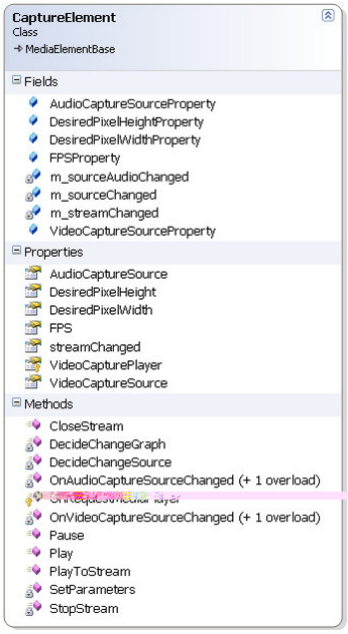


Figura 6: Vista completa de clase CaptureElement.

```
private void DecideChangeGraph(string url, string fileUrl)
{
   if (m_streamChanged)
   {
       string videoSource = VideoCaptureSource;
       string audioSource = AudioCaptureSource;
       VideoCapturePlayer.Dispatcher.BeginInvoke((Action)delegate
       {
           VideoCapturePlayer.changeToStream = true;
           VideoCapturePlayer.streamURL = url;
           VideoCapturePlayer.fileURL = fileUrl;
       });
       m_streamChanged = false;
   }
}
```

Código 12: Implementación de DecideChangeGraph.

```
/// <summary>
/// Begin Streaming to WMS method
/// </summary>
public void PlayToStream(string url, string fileUrl)
{
   DecideChangeGraph(url, fileUrl);
   base.Play();
}
```

Código 13: Implementación del PlayToStream.

En los códigos anteriores exhiben a modo de ejemplo algunos de los métodos implementados. El miembro privado m_streamChanged es el que se agrega para modelar el control de cambio de estado antes mencionado durante el desarrollo de la entrevista.

VideoCapturePlayer

Esta clase, ya provista por el paquete MediaKit original, es el reproductor que incluye dicho kit para presentar al usuario final el flujo capturado.

Considerando la reestructura diseñada a nivel de DirectShow para soportar las nuevas funcionalidades requeridas en la emisión de la entrevista se debió modificar esta clase de forma de reflejar los cambios introducidos. Es aquí donde se construye y ejecuta el grafo DirectShow de captura/emisión descrito detalladamente en el capitulo 2 de este anexo.

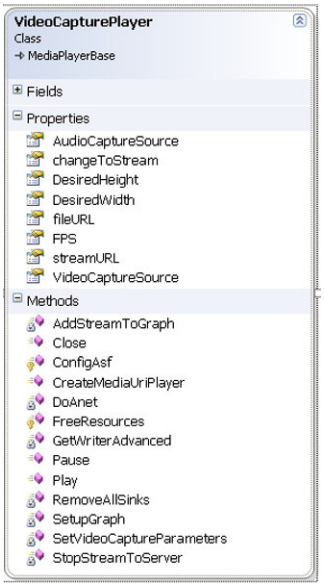


Figura 7: Vista completa de clase VideoCapturePlayer.

En los métodos SetupGraph y AddStreamToGraph se construye el grafo en sus dos etapas de la entrevista, mientras que ConfigAsf, RemoveAllSinks, GetWriterAdvanced y DoANet son métodos auxiliares incorporados que permiten el formateo del flujo con WMF. También debieron ser modificados los métodos de control Play y Close.

```
/// Plays the video capture device
public override void Play()
{
VerifyAccess();

if(m_graph == null)
                    SetupGraph();

if (m_changeToStream){
                    base.Stop();
                    AddStreamToGraph(m_graph);
}
   base.Play();
}
```

Código 14: Modificación en la implementación de método Play.

En el código 14 se puede ver la lógica en la implementación del método Play, donde a su vez se invocan los métodos Play/Stop implementados en su clase base. Esta clase es la que permite, entre otras cosas, manejar el control de multimedia en el grafo a través de la interfaz IMediaControl.

Finalmente, se debió implementar un método que lograra cerrar la sesión de comunicación abierta con el servidor WMS que permanece activa mientras se realiza la emisión al punto de publicación.

```
private void StopStreamToServer()
{
if (m_pWriterAdvanced != null)
{
       m_pWriterAdvanced.RemoveSink(m_pPushSink);
      m_pPushSink.EndSession();
      Marshal.ReleaseComObject(m_pWriterAdvanced);
      m_pWriterAdvanced = null;
   }
   if (m_pPushSink != null)
   {
       Marshal.ReleaseComObject(m_pPushSink);
       m_pPushSink = null;
   }
```

Código 15: Implementación de StopStreamToServer.

Propiedades

Los controles personalizados en WPF son definidos por medio de la inserción de una directiva que se inserta en el código XAML.

Este es el caso de MediaKit que como se dijo presenta un conjunto de controles personalizados para el manejo de multimedia.

```
xmlns:DShow="clr-namespace:WPFMediaKit.DirectShow.Controls;assembly=WPFMediaKit"
```

Figura 8: Directiva de definición para MediaKit en WPF

Los controles de WPF constan de propiedades por medio de los cuales se configura su comportamiento y características.

Existen dos formas de definir controles y configurar sus propiedades. La primera es por medio de un bloque en el archivo XAML.

```
<DShow:CaptureElement x:Name="videoCapElement"
             LoadedBehavior="Play"
             DesiredPixelWidth="320"
             DesiredPixelHeight="240"
             Stretch="Fill"
             VideoCaptureSource="{Binding Path=SelectedItem,
                                  ElementName=videoCapDevices}"

             AudioCaptureSource="{Binding Path=SelectedItem,
                                  ElementName=audioCapDevices}"
             FPS="30" />
```

Código 16: Definición de control CaptureElement en XAML.

En el siguiente código se muestra la forma en que se puede configurar el control a través de la codificación en lenguaje C sharp de .NET. A modo de ejemplo se muestra la definición del elemento CaptureElement usado en la aplicación.

```
//Inicia captura de camara MediaKit
videoCapElement = new WPFMediaKit.DirectShow.Controls.CaptureElement();

//Setea parametros de captura
videoCapElement.LoadedBehavior = WPFMediaKit.DirectShow.MediaPlayers.MediaState.Play;
videoCapElement.DesiredPixelWidth = 320;
videoCapElement.DesiredPixelHeight = 240;
videoCapElement.Stretch = Stretch.Fill;
videoCapElement.VideoCaptureSource = (string)videoCapDevices.SelectedItem;
videoCapElement.AudioCaptureSource = (string)audioCapDevices.SelectedItem;
videoCapElement.FPS = 30;
videoCapElement.VideoCaptureSource = (string) videoCapDevices.SelectedItem;
videoCapElement.AudioCaptureSource = (string) audioCapDevices.SelectedItem;
WinMainEmisionStack.Children.Add(videoCapElement);
```

Código 17: Declaración de control en C sharp.

La diferencia entre ambas está dada por el momento en que se crea el objeto definido. Por este motivo en la solución de captura presentada, se inicia y carga este control en el momento de ser usado, por lo que se optó por implementar por programación en C sharp.

3.2 Audio con DirectSound

La captura de audio y video desde los dispositivos de entrada es realizada por medio de la librería DirectShow.

A los efectos de brindarle al usuario final una noción sobre cómo está saliendo el audio emitido, se presenta en la aplicación un medidor que permite monitorear gráficamente el nivel de audio en todo momento.

Esta funcionalidad se implementa en un hilo de ejecución adicional, que a través de la librería DirectSound y haciendo uso de un control progressBar -que hace las veces de medidor- resuelve la necesidad planteada.

La manera de conseguir información durante la captura en dispositivos de audio es por medio de un buffer implementado en DirectSound que permite registrar cada muestra capturada.

En una primera instancia se le debe dar formato al contenido del buffer, para luego dar comienzo a la captura de muestras.

```
WaveFormat wf = new WaveFormat();
wf.BitsPerSample = 16;
wf.SamplesPerSecond = 44100;
wf.Channels = 2;
wf.BlockAlign = (short)(wf.Channels * wf.BitsPerSample / 8);
wf.AverageBytesPerSecond = wf.BlockAlign * wf.SamplesPerSecond;
wf.FormatTag = WaveFormatTag.Pcm;


CaptureBufferDescription desc = new CaptureBufferDescription();
desc.Format = wf;
desc.BufferBytes = SAMPLES * wf.BlockAlign;
buffer = new DirectSound.CaptureBuffer(desc, cap);
buffer.Start(true);
```

Código 18: Configuración del formato de la muestra e inicio de captura.

Es así que, cada cierta cantidad de milisegundos se procesa el contenido del buffer promediando los valores de las últimas 8 muestras capturadas obteniendo el valor de amplitud a graficar.

```
/* Obtengo la suma de los ultimos 8 samples
 * capturados en el buffer. */
 for (int i = 0; i < SAMPLES; i++)
 {
   sum += (Int16)samples.GetValue(i, 0, 0);
 }

 /* Calcula el valor absoluto de
 * la amplitud promedio */
 double average = Math.Abs(sum / SAMPLES);
```

Código 19: Obtención de amplitud promedio en instante x.

La unidades de volumen (VU meters) son logarítmicas por naturaleza, así que resulta importante hacer una buena elección del rango de valores a mostrar. Por ejemplo, la entrada de micrófono en un PC suele tener un ruido piso de -80dB, por lo que si se elige representar una escala de -120dB, entonces una gran porción del medidor será ruido.

Por último, para lograr esta representación logarítmica y así construir la unidad de monitoreo de volumen necesitada por la aplicación se debe convertir cada valor de amplitud calculada a su correspondiente en la escala decibeles.

```
const double MIN_DB = -34.5;
const double MAX_DB = 0;

 /* Convierte el valor promedio a dB */
double dB = (average > 0) ? 20 * Math.Log10(average / 32768) : -1000000;
double am = Math.Max((dB - MIN_DB) * 100 / (MAX_DB - MIN_DB), 4);
```

Código 20: Fórmula para pasaje de muestras promedio a dB.

3.3 Captura imagen para portada

En el momento de comenzar la emisión de una entrevista se registra una captura de imagen que es grabada en formato JPG. Esta imagen tomada es la denominada imagen de portada y que posteriormente formará parte de la entrevista.

Para realizar esta operación se graba un nuevo archivo por medio de un stream que utiliza las clases RenderTargetBitmap, JpegBitmapEncoder y BitmapEncoder.

```
// captura de imagen para portada
int Wi= (int)videoCapElement.ActualWidth;
int He= (int)videoCapElement.ActualHeight;
RenderTargetBitmap bmp=new RenderTargetBitmap(Wi,He,96,96,PixelFormats.Pbgra32);
bmp.Render(videoCapElement);

// setea el nombre de la imagen .jpg
string file= Transforma.FormatoURL(path, idEntrevista, Formato.Imagen);
BitmapEncoder encoder= new JpegBitmapEncoder();
encoder.Frames.Add(BitmapFrame.Create(bmp));
using (Stream stm= File.Create(file))
{
     encoder.Save(stm);
     stm.Close();
     stm.Dispose();
}
```

Código 21: Creación de imagen JPG.

4. Presentación de emisión en vivo en WPF

La emisión en vivo puede ser presenciada por un grupo de usuarios invitados que deben apuntar a una dirección de publicación presentada por el servidor.

Para reproducir dicho flujo multimedia la aplicación cuenta con un objeto MediaElement de WPF. A través de la propiedad Source se configura la dirección URL de donde el cliente debe obtener el flujo emitido.

Esta dirección se forma por la concatenación de un moniker –RTSP o MMS tratando de aprovechar la mejor condición de transmisión-, la dirección del servidor de streamig, el puerto y el punto de publicación correspondiente.

Cuando una URL especifica por ejemplo *mms://*, el objeto reproductor intenta usar, en ese orden, los siguientes protocolos para entrega de datos.

1- RTSPU (RTSP usando UDP)
2- RTSPT (RTSP usando TCP)
3- MMSU (MMS usando UDP)
4- MMST (MMS usando TCP)
5- HTTP

El HTTP es un protocolo unidireccional basado en TCP y usado en todos los servidores Web. El streaming a través de HTTP es mucho menos eficiente que usar RTSP, sin embargo muchos firewalls están configurados para aceptar tráfico HTTP, rechazando otros protocolos de streaming lo que permite una viabilidad de la solución en estos entornos.

5. Presentación de entrevista grabada en WPF

De manera similar a la emisión en vivo se usa el control MediaElement de WPF para reproducir a demanda los videos descargados. Esta vez, la propiedad Source se define con el camino absoluto en disco local y el nombre completo del archivo multimedia. En este caso, la reproducción del video debe estar sincronizada con la aparición de mensajes escritos en el chat por lo que se usa un hilo adicional que controla estas tareas.

6. Referencias

- DirectShow – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/library/dd375454(VS.85).aspx
- DirectShow and Windows Media – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/library/dd757032(VS.85).aspx
- Extracto de artículo "Industry for measuring audio levels", Sound Pro Magazine – último acceso: 30/06/09
  http://web.onetel.net.uk/~peterphmusic/pharri5833/metering.htm
- Reading ASF Data Over a Network – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/library/dd757588(VS.85).aspx
- Sending ASF Data Over a Network – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/library/dd743729(VS.85).aspx
- Using the Callback Methods – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/library/dd743944(VS.85).aspx
- Using the OnStatus Callback – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/library/dd743950(VS.85).aspx
- Working with Profiles – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/library/dd758039(VS.85).aspx
- Writing ASF Files – último acceso: 30/06/09
  http://msdn.microsoft.com/en-us/library/dd758048(VS.85).aspx

2. Multimedia en Windows

1. Librerías DirectX

Se le denomina DirectX a un conjunto de interfaces de programación (APIs) que permiten facilitar el control de tareas multimedia, mediante un acceso directo al hardware.

DirectX está especialmente enfocada al diseño, construcción y programación de juegos y video para plataformas Windows e incluye librerías para el desarrollo de dichas aplicaciones (Software Development Kit SDK) que pueden ser descargadas gratuitamente desde el sitio oficial de Microsoft Windows.

Si bien estas librerías fueron diseñadas y desarrolladas exclusivamente para la plataforma Windows, existe actualmente una avanzada implementación de su API para sistemas tipo Linux llamada Cedega.

1.1 Historia

Antes de la existencia de DirectX los desarrolladores realizaban sus aplicaciones en MS-DOS, debiendo programar explícitamente, a través del sistema operativo, la detección y el acceso directo a los distintos dispositivos de hardware que tenía el computador. Con el paso del tiempo las tendencias marcaron un crecimiento importante en la diversidad de componentes de hardware lo que dificultaba aún más la tarea de los programadores.

La salida de Windows95 ofreció a los usuarios muchos avances. La incorporación de características del estilo "Plug and Play" permitiendo una fácil instalación del hardware o la programación multitarea lograron mejorar el rendimiento y la relación usuario-máquina, pero aún así, el acceso directo al hardware estaba protegido por el sistema operativo, por lo que acceder a tarjetas gráficas o de sonido era lento e ineficaz. En noviembre de 1994, Craig Eisler, Alex St. John y Eric Engstrom, se plantearon convertir a Windows95 una plataforma en la que los juegos pudieran acceder a las altas prestaciones de las tarjetas gráficas en ese entonces existentes.

Fue así que Microsoft creó una API llamada The Game SDK que permitió el desarrollo efectivo de aplicaciones CAD, audio y video, que poco más tarde se renombró como DirectX. Las primeras versiones no resultaron tan claras y eficientes como se esperaba, por lo que recién a partir de de versión 5.0 se dio el salto definitivo, logrando una API más estable, rápida y eficiente.

En el antiguo MS-DOS la comunicación entre la aplicación y el hardware se realizaba directamente, mientras que en Windows95 debía realizarse forzosamente a través del sistema operativo.

Un sistema con DirectX, permite el acceso al hardware a través de la propia API DirectX sin tener que pasar por el sistema operativo, es decir DirectX hace de puente entre aplicación y hardware.

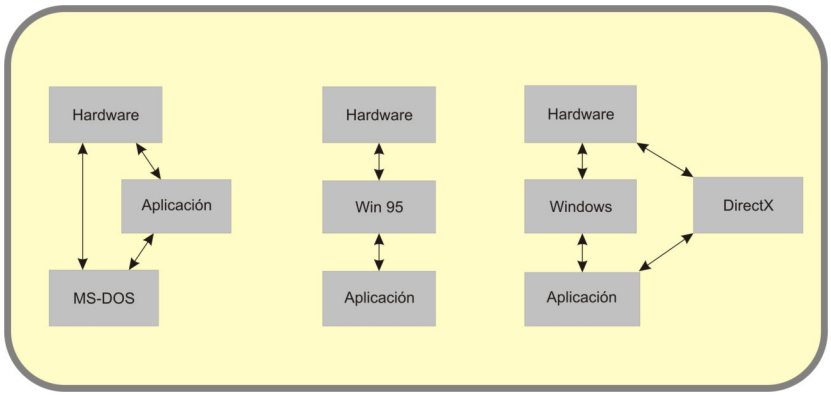


Figura 1: Estructuras de comunicación entre aplicación y hardware.

Actualmente la última versión de DirectX, publicada en enero de este año, es la versión 11 desarrollada para Windows7.

1.2 Componentes

El término DirectX agrupa un conjunto de APIs en la que cada una es enfocada a ciertas tareas específicas. El siguiente es un detalle de las distintas componentes que contiene DirectX.

- Direct3D, procesa y programa gráficos en tres dimensiones.
- DirectX Graphics, gestiona objetos en dos dimensiones (imágenes planas), y actúa además como administrador de la memoria de video.
- DirectInput, procesa los datos que llegan de los dispositivos de entrada como el teclado, el ratón o el joystick.
- DirectPlay, conjunto de herramientas que simplifican el acceso a los servicios de comunicación, dando soporte a los programas que se ejecutan en red. Como conexiones entre distintos usuarios, sesiones o mensajes entre usuarios. Aunque está enfocada principalmente a los juegos, también puede ser usada por cualquier aplicación.
- DirectSound, se utiliza para desarrollar aplicaciones de audio de alto rendimiento. Puede reproducir varios formatos de audio, así como dar salida a varios sonidos a la vez o capturarlos desde dispositivos de entrada. Además, administra la memoria de la tarjeta de sonido. Sólo utiliza audio digital.
- DirectMusic, se encarga de reproducir archivos de sonido y pistas musicales.
- DirectShow reproduce y captura archivos multimedia.
- DirectSetup, para la instalación completa o personalizada de componentes DirectX.

1.3 Funcionamiento interno

El principal cometido que busca DirectX es que las aplicaciones multimedia puedan alcanzar un rápido acceso al hardware y que estas sean, a su vez, independientes a los dispositivos instalados.

Con el fin de lograrlo DirectX utiliza una librería de rutinas que se comunican directamente con los drivers del hardware.

Se dispone de dos interfaces a través de las cuales se realizan las peticiones a los dispositivos, la capa de abstracción de hardware (HAL) y la capa de emulación de hardware (HEL).

La interfaz HAL es la que los fabricantes de adaptadores gráficos deben ofrecer a DirectX. En esta capa se implementa todo el código de bajo nivel relacionado al dispositivo, código que accederá directamente al hardware gráfico aprovechando al máximo sus posibilidades, dado que es escrito por el propio fabricante.

En aquellos casos que el hardware no implementa determinada funcionalidad necesaria para DirectX la capa HAL lo detecta y en ese momento se utiliza la capa HEL que, mediante instrucciones de software, realiza aquellas operaciones no disponibles en el hardware.

De esta manera no es necesario escribir código específico para cada tipo de hardware, solamente se debe desarrollar la aplicación para que DirectX se encargue de traducir la aplicación al hardware mediante sus drivers.

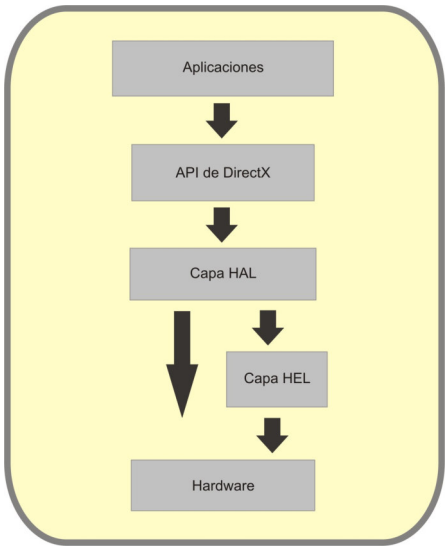


Figura 2: Capas HEL y HAL.

La figura 2 ilustra el comportamiento descrito, donde cada una de las llamadas a los servicios de DirectX es directamente traducida y enviada al hardware, por la capa HAL, o bien simulada por software, por la capa HEL.

2. Librerías DirectShow

DirectShow es una API que le brinda a las aplicaciones de Windows la posibilidad de controlar una amplia variedad de dispositivos de entrada de audio y video, incluyendo, cámaras de video, cámaras Web, lectoras de DVD, tarjetas de sintonización de televisión, entre otros. Además, proporciona soporte a una gran variedad de formatos, desde WAV o AVI hasta el propio formato Windows Media. Esta API también es extensible dado que permite a terceros dar soporte a sus propios dispositivos, formatos y componentes de proceso especializados.

Una de las mejores características de Microsoft Windows XP es la mejora en el manejo de audio y video digital. Esto es sumamente notorio en los avances que han tenido tanto Windows Media Player como Movie Maker.

La tecnología DirectShow es una poderosa arquitectura de streaming diseñada para el manejo de datos de audio y video soportado por prácticamente todos los dispositivos de medios digitales que funcionan en Windows.

DirectShow ofrece un modelo de aplicación de alto nivel que permite un desarrollo rápido de las aplicaciones para medios digitales. También cuenta con un modelo plug-in de bajo nivel que habilita a terceros a crear componentes de procesamiento de datos multimedia a medida.

2.1 Evolución de DirectShow

La primera forma que tuvo Microsoft Windows de capturar video fue a través de Video para Windows (VfW, por sus iniciales en inglés) para Windows 3.1. VfW era bastante bueno para su época, pero tenía algunas limitaciones. Una de las más importantes era el hecho de que el Manejador de Compresión de Video (VCM, su sigla en inglés), no estaba diseñado para manejar codecs que, durante el proceso de compresión, colocan cuadros de video en diferente orden. Dicha limitación trae como consecuencia la dificultad de escribir codecs MPEG en base a VCM.

Windows entonces, recurrió la Interfaz de Control de Medios (MCI) y el uso de drivers mciavi, para ofrecer una alternativa al manejo de reproducciones simples de audio y video. A

pesar de que la infraestructura de la MCI permitía decodificadores MPEG, nunca fue totalmente trasladada a la arquitectura de 32 bits.

Para hacer frente a estas limitaciones, Microsoft comenzó un proyecto conocido como Quartz, cuyo propósito inicial fue proveer soporte para reproducir archivos MPEG-1 en Windows.

Ante la multitud de nuevos dispositivos tales como videocámaras, nuevos formatos para medios como el MPEG-2, el DVD, y nuevas tecnologías como las videoconferencias se puede ver que lo que realmente se precisa es un framework en el que todas esas tecnologías puedan trabajar juntas eficientemente. Además, este framework debería prever que otros puedan brindar soporte a hardware especial, formatos no estándar y aplicaciones para procesar operaciones. El framework debería simplificar al máximo el desarrollo de aplicaciones, y a la vez seguir teniendo la posibilidad de controlar operaciones de streaming de bajo nivel y modificar datos de multimedia cuando sea necesario.

El equipo Quartz tomó un proyecto ya existente llamado Clockwork para enfrentar la ampliación de estos requerimientos. Clockwork era un framework modular en el que componentes semi independientes trabajaban juntos bajo reglas preestablecidas buscando procesar streams de medios digitales. El resultado fue una arquitectura de streaming basada en COM que desde hace tiempo ha servido como base para el desarrollo de aplicaciones para Windows.

Esta arquitectura originalmente se llamó ActiveMovie que, como ya se dijo, en 1995 fue lanzada con el nombre de DirectX Media SDK.

En el año 1996, ActiveMovie pasó a llamarse DirectShow. En 1998, con la salida de DirectX Media 6.1, se agregó soporte para DVD y aplicaciones de televisión análoga. Sin embargo, no fue sino hasta el año 2000 con DirectX 8.0, que DirectShow pasó a formar parte de DirectX SDK. Fue en esta etapa que se agregaron soporte para Windows Media Format (WMF) y una API para la edición de video en DirectShow.

2.2 Filtros y streams multimedia

Los streams de video digital son secuencias de cuadros o frames de video que pueden ser mapas de bits en RGB o, si el streaming está comprimido, un conjunto de valores numéricos que habilita a un decodificador a reconstruir la imagen final. El contenido exacto de la información varía dependiendo del formato de video que se trate. En todos los formatos, el video en movimiento típicamente es reproducido a 25 o 30 cuadros por segundo. Los streams de audio digital descomprimido consisten en una secuencia de muestras, cada una de las cuales es un entero que representa la amplitud (redondeada) de una señal análoga en un determinado momento.

Para calidad de audio de CD, las muestras tienen una precisión de 16 bits que luego son grabadas y reproducidas a 44.1KHz. Un stream de audio comprimido no contiene las secuencias de muestras reales, pero como con los streams de video, contiene valores de decodificación que usa para reconstruir el stream original antes de pasarlo por la tarjeta de sonido. Además de la compresión y descompresión, que son realmente necesarios si se considera la limitada capacidad de almacenamiento y ancho de banda, los streams de audio y video digital pueden ser procesados de distintas e interesantes formas. Los streams digitales pueden ser combinados, analizados, reorganizados, copiados, generados y modificados en forma mucho más sencilla que en el mundo analógico.

En DirectShow, todas las operaciones de stream están encapsuladas como filtros. Estos son objetos COM que tienen comportamientos estándar acorde a las capacidades puntuales que se les haya dado en cada caso. Tanto las lectoras, multiplexores, compresores y descompresores, así como los propios drivers de los dispositivos, son filtros que saben cómo comunicarse con –y enviar datos a- otros filtros. Las aplicaciones son construidas conectando estos filtros unos a otros para desarrollar una determinada tarea.

Existen tres tipos básicos de filtros: filtros de origen o filtros source para entrada de datos, filtros transformadores para pasos de procesamiento intermedio, y filtros de entrega para presentar la salida de datos.

Un filtro de origen introduce información al stream. Estos datos pueden originarse en un archivo, en un dispositivo tal como una cámara de video, cámara Web, sintonizador de televisión o cualquier tipo dispositivo existente. DirectShow está acoplado al Modelo de Drivers de Windows (WDM), por lo que cualquier dispositivo con un WDM correctamente implementado está expuesto automáticamente, a través de los filtros de origen de DirectShow, a las aplicaciones en el modo usuario. DirectShow también proporciona filtros de origen para el ingreso de datos desde archivos, DVDs y otros dispositivos VfW.

Los filtros transformadores o de transformación reciben datos desde otro filtro, realizan alguna operación en la información y la pasan a otro filtro de stream de bajada. Las transformaciones pueden ser análisis del stream, codificar o decodificar, incrustación de texto, o cualquier tipo de análisis o manipulación de los bits de audio o video. DirectShow ofrece muchos filtros transformadores para manejar varios formatos de archivos y de compresión, incluyendo señales analógicas y digitales de televisión.

Un filtro de entrega o render acepta datos provenientes tanto desde un filtro de origen, como de uno transformador y permite su salida por la pantalla, los parlantes, un archivo o un dispositivo.

El término "Direct" en DirectShow refleja el hecho que este tipo de filtro utiliza la tecnología de DirectDraw y DirectSound para pasar los datos de modo eficiente a la tarjeta gráfica y de video. Además, DirectShow tiene la capacidad de realizar streaming a nivel de núcleo del sistema, lo que le permite capturar dispositivos como sintonizadores de televisión o drivers para DVD que pasan información a dispositivos de salida directamente en modo monitor. De esta forma se ahorra el costo de transición entre los modos usuario y monitor en situaciones en las que la aplicación no requiere de ellos.

Con la división de tareas dentro de los conjuntos de filtros independientes se optimiza la reutilización de código. Por ejemplo, en un escenario de reproducción de un archivo, habría dos operaciones en común: leer el dato puro desde el stream y mandar el resultado final para la tarjeta gráfica o de sonido. No importa cuál sea el formato, siempre se puede usar el mismo filtro de DirectShow para realizar estas tareas. Lo único que cambiaría serían los filtros intermedios.

2.3 Modelo de aplicación

DirectShow brinda la capacidad de controlar operaciones de bajo nivel, simplificando el desarrollo al ubicar virtualmente todo el streaming y el procesamiento de multimedia en filtros individuales. La principal función de una aplicación es poner esos filtros a trabajar juntos. Para esa tarea las aplicaciones usan un objeto de alto nivel llamado Manejador del Grafo (Filter Graph Manager, FGM). Este objeto se utiliza para interconectar filtros, cualquiera sea la combinación requerida para desarrollar la tarea asignada. Las tareas pueden ser tanto una sencilla reproducción, como conversiones de formato, análisis de video o incluso corrección de color. La colección de filtros interconectados se llama grafo de filtros (graph filter). Los datos en un grafo siempre fluyen desde filtro de origen a uno de entrega.

Por medio de un FGM se pueden agregar filtros en forma individual, o usar la lógica de conexión inteligente dentro del FGM para automatizar el proceso de construcción del grafo. Al utilizar está lógica de conexión, se construyen grafos de filtros más complejos con solo un par de llamadas a métodos. Una vez el grafo está construido, la aplicación controla sus operaciones en el FGM con los métodos Play, Stop y Pause, que manejan los detalles de sincronización a bajo nivel. Así, el FGM comunica eventos desde el grafo de filtros a la aplicación.

Como ya se mencionó, DirectShow brinda la posibilidad de controlar operaciones de streaming a bajo nivel exponiendo interfaces COM en filtros individuales. Comúnmente las aplicaciones usan estas interfaces para configurar un filtro antes de que comience el streaming.

En el caso que se quiera reproducir un archivo determinado, dado su nombre, el FGM revisa la extensión del archivo, así como los bits reales del stream para verificar el formato del mismo. Posteriormente, busca en el registro un filtro asociado con ese tipo de formato y si encuentra uno, entonces lo agrega al grafo. Después, el FGM examina el tipo de salida de dicho filtro y vuelve al registro buscando filtros que puedan aceptar este formato como entrada de datos. Este proceso se repite hasta que se encuentra un filtro de entrega, momento en el que se conecta. Por defecto, se seleccionará un filtro de entrega que corresponda a la tarjeta gráfica o de sonido, pero si se quiere grabar la salida en un archivo, entonces se puede simplemente agregar al grafo un filtro tipo file-writer.

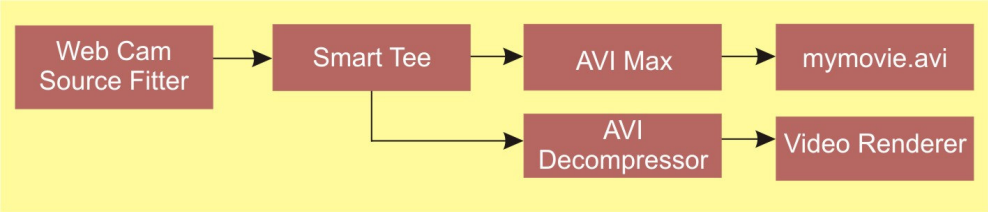


Figura 3: Grafo para captura de cámara web.

DirectShow también proporciona objetos que permiten construir tipos de grafos mucho más complejos. Los grafos pueden obtener los datos a partir de una fuente en vivo, tales como una cámara o un sintonizador de televisión, y salvar el stream en disco o visualizarlo en una pantalla. La figura 3 muestra un esquema de grafo en el que se captura video en vivo con una cámara web, escribe lo capturado en un archivo AVI, y muestra una vista previa. Este grafo es muy similar a la reproducción antes descrita, pero ahora se usa un dispositivo de hardware como fuente, por lo que se deberán usar algunos objetos de DirectShow para localizar el dispositivo y construir el grafo.

2.4 Tipos de multimedia (MediaType)

Otro de los principales beneficios de DirectShow es su extensibilidad. El diseño modular de DirectShow requiere que todos los filtros sigan determinadas reglas para comunicarse con otros filtros. Estas reglas generan los protocolos de conexión y streaming. Antes de que los filtros puedan comunicarse a través de estos protocolos, es necesario que hablen un lenguaje común. Este lenguaje es el MediaType, un grupo de estructuras de datos que colectivamente describen los datos en el stream.

La estructura descriptiva de más alto nivel es el AM_MEDIA_TYPE.

```
typedef struct _MediaType {
        GUID        majortype;
        GUID        subtype;
        BOOL        bFixedSizeSamples;
        BOOL        bTemporalXCompression;
        ULONG       lSampleSize;
        GUID        formattype;
        IUnkown     *pUnk;
        ULONG       cbFormat;
        BYTE        *pbFormat;
        } AM_MEDIA_TYPE;
```

Figura 4: Estructura de AM_MEDIA_TYPE.

Los primeros dos campos, majortype y subtype, son optimizaciones usadas durante el protocolo de conexión para identificar un tipo de stream. En una primera instancia son usados por el FGM en su lógica de conexión inteligente y por los filtros para negociar una conexión. Los tres GUIDs majortype primarios son Audio, Video y Stream. Los majortype Audio y Video auto describen su tipo, mientras que Stream indica que se trata de un tipo desconocido hasta el momento, o un stream de datos compuesto por múltiples streams elementales de diferentes tipos.

Los subtypes brindan un mayor nivel de detalle a los filtros y son usados para determinar rápidamente si un filtro es capaz de hacer lo requerido.

El miembro lSampleSize especifica el tamaño de cada muestra multimedia. Generalmente, una muestra de audio contiene entre 200 y 2000 milisegundos de datos, y una muestra de video contiene un cuadro entero. Si el campo bFixedSizeSamples es falso, se está indicando que el stream está comprimido y por tanto que no todos los cuadros son del mismo tamaño.

El miembro bTemporalCompression señala que el stream de datos está comprimido usando un esquema de compresión inter cuadro, como lo es MPEG.

La información realmente interesante sobre un stream multimedia está en el campo pbFormat, que es un puntero a un boque de formato. El campo formattype identifica el tipo de estructura de datos al que se apunta, y el cbFormat indica el tamaño de la estructura en bytes.



Figura 5: Estructura Info de video.

```
typedef struct  tagVIDEOINFOHEADER {
        RECT            rcSourse;
        RECT            rcTarget;
        DWORD           dwBitRate;
        DWORD           dwBitErrorRate;
        REFERNCE_TIME     AvgTimePerFrame;
        BITMAPINFOHEADER bmiHeader;
        } VIDEOINFOHEADER;
```

Figura 6: Estructura de VIDEOINFOHEADER.

Como se puede apreciar, un VIDEOINFOHEADER es en esencia un BITMAPINFOHEADER con campos adicionales que especifican determinados parámetros de streaming. Esto refleja el hecho de que un stream de video es una secuencia de bitmaps independientes que son mostrados a una determinada velocidad, generalmente de entre 10 a 30 cuadros por segundo. Los rectángulos rcSource y rcTarget permiten a los filtros adaptar los bitmaps de video entrante según la necesidad. El BITMAPINFOHEADER, especifica el ancho y altura del rectángulo, la cantidad de bits por píxel, así como el tipo de compresión, si se está usando alguna. Para los streams de audio, la estructura WAVEFORMATEX proporciona el mismo tipo de información que la descrita aquí para video.

2.5 El Protocolo de conexión

Antes de que dos filtros puedan intercambiar stream de datos, deben usar el lenguaje MediaType para acordar el formato de dicho intercambio. Los filtros logran esto por medio de un objeto llamado pin. Cada vez que un filtro debe conectarse a otro, lo hace a través de un pin. Existen dos variedades de pines: de entrada y de salida

La aplicación, por medio del FGM, es la responsable de agregar los distintos filtros al grafo, y por tanto de especificar a cuál de los pines deben conectarse los mismos. Cuando dos pines se conectan, deben llegar a un acuerdo respecto al formato de la información a transferirse, cómo esta se intercambiará y un allocator, que será el encargado de crear y manejar los buffers de almacenamiento.

El protocolo de conexión comienza en el momento que el FGM llama IPin::Connect en el pin de salida. En este momento, el FGM le pasa un puntero al pin de entrada y, opcionalmente, un puntero a la estructura AM_MEDIA_TYPE. El pin de salida examina el majortype y el subtype para determinar si vale la pena continuar con el proceso de conexión.

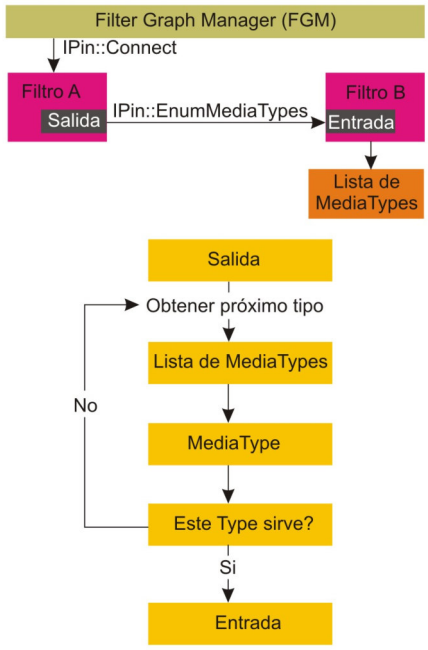


Figura 7: Protocolo de conexión de filtros.

En el caso que el tipo MediaType sugerido sea aceptable, el pin de salida llamará al método IPin::EnumMediaTypes para conseguir una lista de los tipos preferidos por el pin de entrada. Luego, el pin de salida busca en esta lista, examinando cada tipo obtenido. Si encuentra alguno apropiado, lo especifica por medio de una llamada al método IPin::ReceiveConnection del pin de entrada. Si ninguno de los tipos resultara aceptable la conexión falla.

Una vez que ambos pines llegan a un acuerdo en cuanto al tipo de datos, deben determinar el transporte. El transporte de datos más común es el representado por la interfaz ImemInputPin.

Luego de definido el transporte, ambos pines están listos para negociar el número y tamaño de buffers de memoria que tendrán que compartir cuando de comienzo el stream. En la mayoría de los casos, el proceso de asignación y manejo de buffers es delegado a un objeto allocator.

El pin de salida es el responsable de seleccionar el allocator y configurarle sus propiedades. Hecho esto, se le notifica al pin de entrada qué allocator y propiedades de buffer fueron seleccionadas. En la mayoría de los pines el allocator asigna memoria en el sistema anfitrión, pero a veces los pines correspondientes a filtros de origen o entrega tienen allocators que crean buffers en los propios dispositivos. Si la negociación de allocator resulta exitosa, el FGM comienza a procesar el siguiente filtro en el grafo. Después de que todos los filtros fueron conectados de esta manera, el grafo está listo para funcionar.

2.6 Protocolo de streaming

En un grafo de filtros, cuando por ejemplo la fuente es una emisión en vivo, la información es inyectada por un filtro de origen. El término inyectar la información significa que el filtro va llenado los buffers de datos tan rápido como puede.

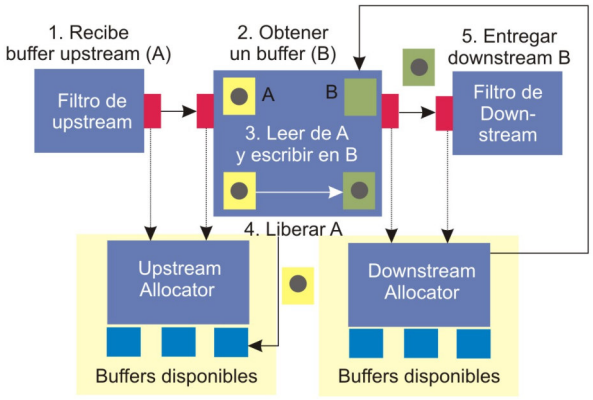


Figura 8: Protocolo de streaming.

Cada bloque de información es encapsulado en un objeto COM que expone la interfaz ImediaSample. Esto se utiliza para almacenar la dirección del buffer y establecer sellos de tiempo en la información. El sello de tiempo indica la hora a la que el cuadro de video o la muestra de audio debería ser entregado al hardware.

Para la mayoría de las operaciones de procesamiento de video en una PC moderna, la capacidad del procesador es muy superior a la necesitada para entregar 30 cuadros por segundo, El desafío aquí, es enlentecer el grafo para que este funcione a la velocidad adecuada. Antes de comenzar, el filtro de entrega inspecciona los sellos de tiempo marcados en las muestras y espera hasta que la hora real coincida con tales marcas para arrancar. Mientras espera, el grafo se bloquea hasta que los buffers estén llenos. El reloj usado para este propósito, generalmente es proporcionado por las tarjetas de sonido, dado que estos son mucho más exactos que el reloj del sistema de la CPU.

Algunas veces no resulta de interés controlar el ritmo del proceso. En esos casos, se puede invocar un método del FGM que le dice al filtro de entrega que no preste atención a los sellos de tiempo y deje correr el grafo tan rápido como pueda. En otras ocasiones, puede que algún filtro que haga un uso intensivo de CPU, tal como un decodificador de video, no esté disponible para seguirle el paso a la velocidad de cuadros determinada. Por esa razón, los decodificadores miran los sellos de tiempo en las muestras que reciben. Si un decodificador recibe una muestra tardía (una cuya hora de presentación ya haya ocurrido), descartará la muestra y seguirá a la siguiente para poder continuar con el streaming del grafo a la velocidad adecuada, y mantener así, el audio y video sincronizados.

Como los filtros de origen y de entrega provistos por DirectShow son adecuados para la mayoría de los escenarios, se tiende a expandir la arquitectura mediante la adición de transformadores personalizados. En este punto, se tienen dos alternativas de implementación: un DirectX Media Object (DMO) o un filtro de DirectShow.

Un DMO es un objeto COM que procesa datos multimedia digitales. Los DMOs son similares a los filtros de DirectShow, pero auto-contenidos. Es decir, no requieren de un grafo de filtros y pueden ser usados fuera de DirectShow. A través de un filtro especial provisto por DirectShow, también se pueden usar DMOs en este entorno, aunque con algunas limitaciones. La decisión de crear un DMO o un filtro está supeditada a las necesidades específicas que se desee cubrir. Definitivamente se recomienda utilizar DMOs para los codificadores y decodificadores, pero en otros escenarios, un filtro es la mejor opción. En general, las interfaces DMOs son más sencillas que las APIs de filtros de DirectShow, aunque las clases bases manejan la mayoría de las complicaciones vinculadas al desarrollo de un filtro.

2.7 Control del flujo

Anteriormente se mencionó el protocolo de streaming que todos los filtros de DirectShow deben soportar. En todo momento, un filtro está en uno de tres estados posibles:

detenido, pausado o funcionando. Cada filtro debe poder alternar entre estos estados en respuesta a los comandos recibidos desde el FGM. Cuando un grafo de filtros se pausa, la muestra se mueve por el grafo hasta que llega al filtro de entrega. En ese punto, el streaming se bloqueará hasta que la aplicación le envíe el comando run. Un filtro de transformación puede no conocer la distinción entre los comandos pause y run, pero en cualquiera de los casos, debe aceptar nuevas muestras procesarlas y enviarlas nuevamente. Cuando el filtro está detenido, debería rechazar muestras nuevas.

3. Librería DirectShowNET

La API DirectShow de Microsoft ha sido diseñada siempre con la mente enfocada en el lenguaje C++. Es por este motivo que el objetivo de la biblioteca DirectShowNET es permitir el acceso a la funcionalidad de Microsoft DirectShow para el desarrollo de aplicaciones multimedia en .NET. La biblioteca es compatible con VisualBasic.NET, C sharp, además de funcionar con cualquier lenguaje .NET compatible. No obstante, las soluciones que permiten acceso a funciones DirectShow obtenidas mediante el uso de esta librería no son tan completas como las que utilizan interfaces desarrolladas para C++.

Esta biblioteca proporciona funcionalidades DirectShow en .NET, conteniendo los enumerados, structs y definición de interfaces para acceder a ellas. Además, es acompañada de un conjunto de ejemplos útiles, que pueden ser descargados del sitio oficial, y sirven para mostrar su forma de uso.

3.1 Interfaces

Aunque cuenta con más de 432 interfaces definidas en el código fuente, no todas han sido testeadas para verificar su correcto funcionamiento.

En la siguiente lista se detallan las interfaces cuyos métodos han sido completamente testeados.

IAMAnalogVideoDecoder
IAMAudioInputMixer
IAMAudioRendererStats
IAMBufferNegotiation
IAMClockAdjust
IAMClockSlave
IAMCopyCaptureFileProgress
IAMCrossbar
IAMDecoderCaps
IAMDirectSound
IAMDroppedFrames
IAMErrorLog
IAMExtDevice
IAMExtendedSeeking
IAMExtTransport
IAMFilterMiscFlags
IAMGraphBuilderCallback
IAMGraphStreams
IAMLine21Decoder
IAMMediaContent
IAMMediaContent2
IAMMediaStream
IAMMediaTypeSample
IAMMediaTypeStream
IAMMultiMediaStream

IAMOpenProgress
IAMOverlayFX
IAMResourceControl
IAMSetErrorLog
IAMStats
IAMStreamConfig
IAMStreamControl
IAMStreamSelect
IAMTimeline
IAMTimelineComp
IAMTimelineEffect
IAMTimelineEffectable
IAMTimelineGroup
IAMTimelineObj
IAMTimelineSplittable
IAMTimelineSrc
IAMTimelineTrack
IAMTimelineTrans
IAMTimelineTransable
IAMTimelineVirtualTrack
IAMTuner
IAMTunerNotification
IAMTVAudio
IAMTVTuner
IAMVfwCompressDialogs

IAMVideoCompression
IAMVideoControl
IAMVideoDecimationProperties
IAMVideoProcAmp
IAMWstDecoder
IAnalogTVTuningSpace
IAsyncReader
IATSCChannelTuneRequest
IATSCComponentType
IATSCLocator
IATSCTuningSpace
IAudioMediaStream
IAuxInTuningSpace
IBaseFilter
IBasicAudio
IBasicVideo
IBasicVideo2
IBDA_AutoDemodulate
IBDA_DeviceControl
IBDA_DigitalDemodulator
IBDA_EthernetFilter
IBDA_FrequencyFilter
IBDA_IPSinkControl
IBDA_IPSinkInfo
IBDA_IPV4Filter
IBDA_IPV6Filter
IBDA_LNBInfo
IBDA_SignalProperties
IBDA_SignalStatistics
IBDA_Topology
IBDAComparable
IBroadcastEvent
ICaptureGraphBuilder2
IChannelTuneRequest
IComponent
IComponents
IComponentsNew
IComponentType
IComponentTypes
IConfigAsfWriter
IConfigAviMux
IConfigInterleaving
ICreateDevEnum
ICreatePropBagOnRegKey
IDeferredCommand
IDMOVideoOutputOptimizations
IDMOWrapperFilter
IDVBCLocator
IDVBSLocator
IDVBSTuningSpace
IDVBTLocator
IDVBTuneRequest
IDVBTuningSpace
IDVBTuningSpace2
IDvdCmd

IDvdControl2
IDvdGraphBuilder
IDvdInfo2
IDvdState
IDVEnc
IDVRGB219
IDVSplitter
IEnumComponents
IEnumComponentTypes
IEnumDMO
IEnumFilters
IEnumMediaTypes
IEnumPins
IEnumStreamBufferRecordingAttrib
IEnumTuningSpaces
IErrorLog
IFileSinkFilter
IFileSinkFilter2
IFileSourceFilter
IFilterChain
IFilterGraph
IFilterGraph2
IFilterMapper2
IFilterMapper3
IFrequencyMap
IGraphBuilder
IGraphConfig
IGraphConfigCallback
IGraphVersion
IIPDVDec
IKsPin
IKsPropertySet
ILanguageComponentType
ILocator
IMediaBuffer
IMediaControl
IMediaDet
IMediaEvent
IMediaEventEx
IMediaEventSink
IMediaFilter
IMediaLocator
IMediaObject
IMediaObjectInPlace
IMediaParamInfo
IMediaParams
IMediaPosition
IMediaPropertyBag
IMediaSample
IMediaSample2
IMediaSeeking
IMediaStream
IMediaStreamFilter
IMemAllocator
IMemAllocatorCallbackTemp

IMemAllocatorNotifyCallbackTemp
IMixerOCX
IMixerOCXNotify
IMixerPinConfig
IMixerPinConfig2
IMPEG2Component
IMPEG2ComponentType
IMpeg2Data
IMpeg2Demultiplexer
IMPEG2PIDMap
IMpeg2Stream
IMPEG2StreamIdMap
IMPEG2TuneRequest
IMPEG2TuneRequestFactory
IMPEG2TuneRequestSupport
IMpegAudioDecoder
IMultiMediaStream
IObjectWithSite
IPersist
IPersistMediaPropertyBag
IPersistStream
IPin
IPinConnection
IPinFlowControl
IPropertyBag
IPropertySetter
IQualityControl
IQualProp
IQueueCommand
IReferenceClock
IRegisterServiceProvider
IRenderEngine
IRenderEngine2
ISampleGrabber
ISampleGrabberCB
ISectionList
ISeekingPassThru
IServiceProvider
ISmartRenderEngine
ISpecifyPropertyPages
IStreamBufferConfigure
IStreamBufferConfigure2
IStreamBufferDataCounters
IStreamBufferInitialize
IStreamBufferMediaSeeking
IStreamBufferMediaSeeking2
IStreamBufferRecComp
IStreamBufferRecordControl
IStreamBufferRecordingAttribute
IStreamBufferSink
IStreamBufferSink2
IStreamBufferSink3
IStreamBufferSource
IStreamSample
ITuner

ITuneRequest
ITuningSpace
ITuningSpaceContainer
ITuningSpaces
IVideoFrameStep
IVideoWindow
IVMRAspectRatioControl
IVMRAspectRatioControl9
IVMRDeinterlaceControl
IVMRDeinterlaceControl9
IVMRFilterConfig
IVMRFilterConfig9
IVMRImageCompositor
IVMRImageCompositor9
IVMRImagePresenter9
IVMRImagePresenterConfig
IVMRImagePresenterConfig9
IVMRMixerBitmap
IVMRMixerBitmap9
IVMRMixerControl
IVMRMixerControl9
IVMRMonitorConfig
IVMRMonitorConfig9
IVMRSurfaceAllocator9
IVMRSurfaceAllocatorEx9
IVMRSurfaceAllocatorNotify9
IVMRVideoStreamControl
IVMRVideoStreamControl9
IVMRWindowlessControl
IVMRWindowlessControl9
IVPBaseNotify
IVPManager
IVPNotify
IVPNotify2

IXml2Dex
IAMCameraControl
IConfigAsfWriter2
IDTFilter
IDTFilterConfig
IETFilterConfig
IEvalRat
IATSCLocator2
IBroadcastEventEx
ICAT
ICCSubStreamFiltering
IDigitalLocator
IDTFilter2
IDTFilter3
IDVB_BAT
IDVB_EIT
IDVB_NIT
IDVB_SDT
IDVB_TOT
IDvbLogicalChannelDescriptor
IDvbSatelliteDeliverySystemDescriptor
IDvbTerrestrialDeliverySystemDescriptor
IFilterGraph3
IGenericDescriptor
IPAT
IPMT
IReferenceClockTimerControl
IStreamBufferConfigure3
ITunerCap
IXDSCodecConfig
IAnalogRadioTuningSpace
IAnalogRadioTuningSpace2
IAuxInTuningSpace2
IKsTopologyInfo

4. Librería MediaKit

WPF es una tecnología que ofrece una amplia infraestructura y potencialidad gráfica para desarrollar aplicaciones más atractivas con mayores funcionalidades de animación, audio y video, entre otras. Esta tecnología proporciona, en forma estándar, manejo de multimedia integrado a través de controles específicos basados en Windows Media Player. Sin embargo y a pesar de ser una buena alternativa para reproducción de multimedia, resulta extremadamente duro representar y manipular grafos DirectShow más complejos y personalizados.

Para incorporar estas nuevas funcionalidades Jeremiah Morril desarrolló una biblioteca de multimedia para WPF que publicó en CodePlex y llamó MediaKit.

Esta librería permite crear, incorporar y controlar de forma sencilla reproductores de audio y video personalizados en WPF para cualquier tipo de grafo DirectShow construido.

Inicialmente solo permitía VMR9 como forma de entregar video, pero en sus últimas versiones se ha incorporado soporte EVR para un mejor rendimiento. También implementa su propia interoperabilidad con Direct3D que permite trabajar con arquitecturas de 32 y 64 bits.

Es posible ver la historia y seguir el avance de esta librería en el blog del autor citado en las referencias del anexo.

4.1 Objetos COM y DispatcherObject

Un aspecto importante a considerar es la forma de comunicación con los objetos COM de DirectShow. Dichos objetos están sujetos a las normas de que rigen la comunicación entre hilos. De esta forma, un objeto creado en Single-Threaded-Apartment (STA) solo puede ser accedido desde el hilo que lo creó, mientras que un objeto declarado en Multi-Threaded-Apartment (MTA) puede ser manipulado desde cualquier otro hilo MTA.

Debido al hecho de que un hilo STA no pueda comunicarse con hilos MTA y viceversa, es que el manejo de objetos COM y la interfaz de usuario en WPF resulta un inconveniente importante.

La tecnología WPF utiliza un hilo STA para ejecutar la interfaz de usuario, donde no es recomendable crear objetos COM de DirectShow, ya que cualquier bloqueo en dichos objetos también bloquearía la ejecución de la interfaz. Para resolver este dilema, se aprovecha la facilidad del DispatcherObject ejecutando el despachador en su propio hilo. Así, se podrá acceder en forma segura a cada objeto COM (COM de DirectShow en este caso) utilizando el Dispatcher->Invoke/BeginInoke de WPF.

4.2 Interfaces y clases base

El diseño de todos los controles incluidos en la librería está basado en MediaElement de WPF, ya que este abarca la mayoría de los usos más comunes en la reproducción de multimedia para esta tecnología.

En la figura se muestra un diagrama parcial de clases de la librería donde se puede apreciar las interfaces y clases principales definidas para encapsular toda la funcionalidad del control FrameworkElement.

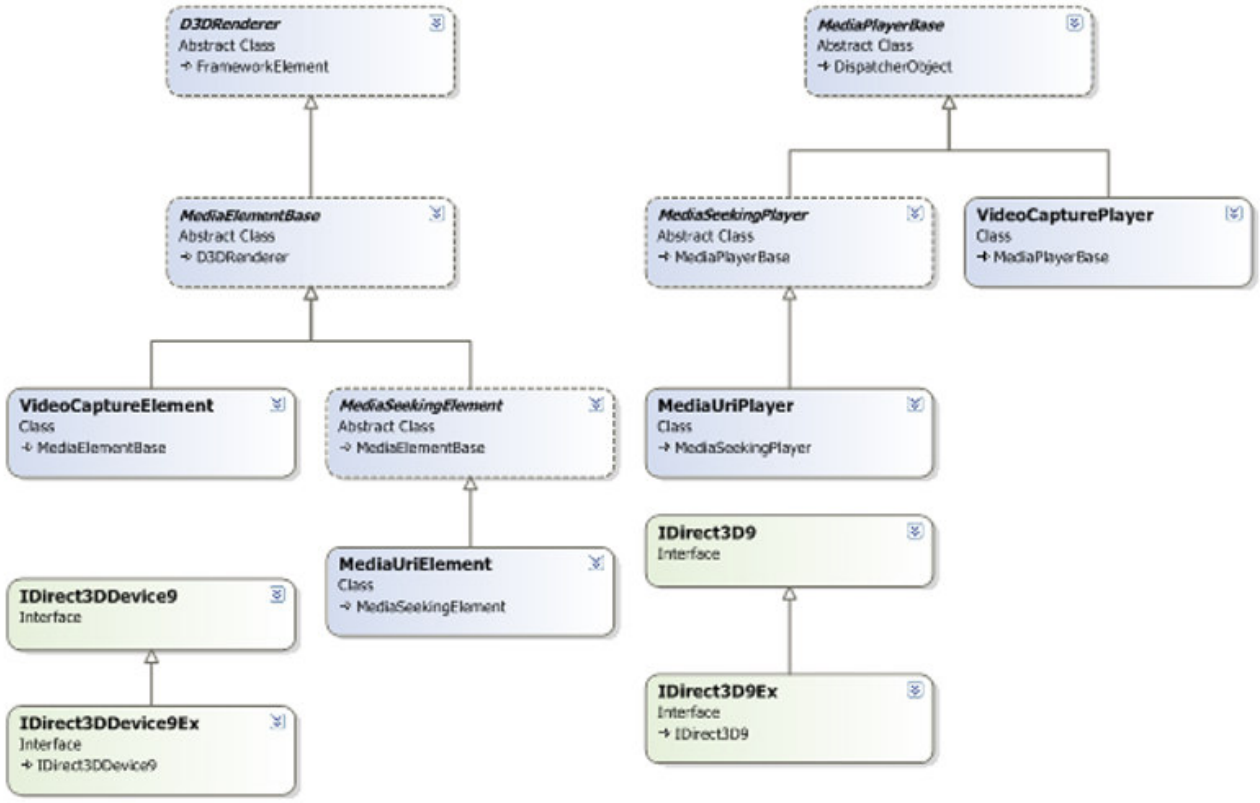


Figura 9: Vista parcial del diagrama de clases de MediaKit.

4.3 Controles

La principal motivación para el desarrollo de MediaKit fue brindarle a WPF la posibilidad de contar con controles personalizados que pudieran agregar más funciones para el manejo de multimedia.

Los controles VideoCaptureElement, MediaUriElement y DvdPlayerElement fueron diseñados en base al MediaElement de WPF de forma tal de que pudieran acoplarse al XAML sin inconvenientes.

Ejemplos de uso de dichos elementos pueden ser vistos en los siguientes códigos.

```
<WPFMediaKit:DvdPlayerElement x:Name="dvdPlayer"
                Stretch="Fill"
                Volume="1"
                PlayOnInsert="true"
                LoadedBehavior="Play"/>
```

Código 1: Objeto DvdPlayerElement en XAML.

```
<WPFMediaKit:VideoCaptureElement x:Name="videoCapElement"
                  LoadedBehavior="Play"
                  DesiredPixelWidth="320"
                  DesiredPixelHeight="240"
                  Stretch="Fill"
                  VideoCaptureSource="{Binding Path=CaptureDeviceName}"
                  FPS="30" />
```

Código 2: Objeto VideoCaptureElement en XAML.

4.4 Últimos avances

La librería, mantenida por su autor, está en continuo desarrollo, por lo que con el transcurrir del tiempo aparecen nuevas funcionalidades y mejoras al código ya existente.

Entre los últimos avances resulta destacable mencionar las mejoras en funciones para reproducción de DVD y los cambios en el tratamiento de las capturas obtenidas desde dispositivos dedicados a este fin.

Las modificaciones al control VideoCaptureElement que permiten capturar desde dos fuentes de video (por ejemplo cámaras Web) y obtener acceso a cada muestra de video capturada son avances que marcan la viabilidad de personalizar estos controles para el desarrollo de *Picture in Picture*.

5. Referencias

- API Cedega – último acceso: 29/06/09
  www.transgaming.com
- Blog del autor de MediaKit – último acceso: 29/06/09
  http://jmorrill.hjtcentral.com/
- DirectShowNET – último acceso: 29/06/09
  http://directshownet.sourceforge.net/
- DirectX – último acceso: 29/06/09
  www.microsoft.com/windows/directx
- Historia de DirectX – último acceso: 29/06/09
  http://craig.theeislers.com/2006/02/directx_then_and_now_part_1.php
- MediaKit – último acceso: 29/06/09
  http://www.codeplex.com/WPFMediaKit

3. Formato de archivos multimedia

1. Introducción

En este anexo se presenta una breve descripción de los formatos y librerías que provee Microsoft Windows para la creación y manejo de archivos multimedia.

Los formatos y sistemas para almacenar o reproducir video digitalizado dependen de algoritmos especiales que controlan la cantidad de información por cuadro de video que se envía a la pantalla, así como la velocidad a la cual se despliegan los nuevos cuadros.

Un video es una sucesión de imágenes presentadas a cierta frecuencia. El ojo humano es capaz de distinguir aproximadamente 20 imágenes por segundo. De este modo, cuando se muestran más imágenes por segundo, es posible engañar al ojo y crear la ilusión de una imagen en movimiento. Además, el video multimedia generalmente está acompañado de sonido, es decir, datos de audio.

El video digital consiste en mostrar una sucesión de imágenes digitales. Dado que estas imágenes digitales se muestran a una frecuencia determinada, es posible saber la frecuencia de refresco, es decir, el número de bytes mostrados o transferidos por unidad de tiempo.

De esta manera, la frecuencia necesaria para mostrar un video -en bytes por segundo- equivale al tamaño de la imagen multiplicado por el número de imágenes por segundo.

Por último, una imagen de video no comprimida ocupa en el entorno de 1MB de espacio. Para obtener un video fluido, se necesita una frecuencia de al menos 25 o 30 imágenes por segundo, lo que genera un flujo de datos de 30 MB/s aproximadamente, es decir, un tamaño de más de 1,5GB por minuto. Resulta claro pensar que esta clase de flujo no es compatible con las capacidades de almacenamiento o incluso con las conexiones de red disponibles actualmente. Por lo tanto, para superar esta dificultad, se puede recurrir a algoritmos (codec) que permitan reducir de forma significativa el flujo de datos por medio de la compresión/descompresión de datos de video.

2. Windows Media Format SDK

El kit de desarrollo Windows Media Format SDK permite a las aplicaciones la posibilidad de acceder a los componentes de Windows Media Format (WMF). Uno de sus principales objetivos es manipular a bajo nivel los distintos componentes de Windows Media. De esta forma, permite a los desarrolladores crear aplicaciones que reproduzcan, graben, editen y hasta codifiquen archivos de formato avanzado (ASF) o fujos multimedia para enviar por la red.

Esta biblioteca también permite la codificación y decodificación de audio o video usando los codecs de Windows Media. Cada codec está optimizado para obtener la mejor calidad en un tipo específico de contenido.

Las características principales de Windows Media Format SDK permiten brindar soporte a funcionalidades de distintos formatos para construír aplicaciones más flexibles.

- Escritura de archivos ASF
  Los archivos se crean sobre la base de perfiles personalizables, lo que permite una fácil configuración y normalización. Es posible escribir archivos de más de 2GB de tamaño.
- Lectura de archivos ASF
  Proporciona soporte para leer tanto archivos locales , como datos que fluyen a través de la red. También presta soporte para muchas características avanzadas de lectura, tales como el soporte nativo para múltiples tasas de bits (MBR). Esta técnica permite la definición de múltiples streams con el mismo contenido codificado a diferentes velocidades. De esta forma, el lector puede seleccionar automáticamente el stream a utilizar, dependiendo del ancho de banda disponible en el momento de la reproducción.

- Entrega de streams ASF a través de la red
  Permite el envío de datos en formato ASF a los equipos remotos a través de HTTP y también soporta la entrega de dichos datos directamente a un servidor remoto de Windows Media (WMS).
- Edición de metadatos en archivos ASF
  La información sobre un archivo y su contenido es fácilmente manipulado por medio de estas librerías.
- Edición de contenido
  Permite que las aplicaciones puedan buscar determinados puntos por tiempo o cuadro dentro de un archivo.
- Lectura y edición de metadatos en archivos formato MP3
  Esta biblioteca proporciona soporte para lectura de archivos MP3 con los mismos métodos utilizados en archivos ASF.

3. Formato Sistema Avanzado (ASF)

El formato avanzado de transmisión por secuencias es el formato de archivo preferido para Windows Media. Si se instalan los codecs apropiados en el equipo, con el reproductor de Windows Media es posible reproducir contenido de audio, de video o ambos que estén comprimidos con una amplia variedad de codecs y que se almacenen en forma de archivo .asf. Además, se puede transmitir por secuencias contenido de audio y video con servicios de Windows Media Services (WMS).

ASF es un formato de archivo extensible diseñado principalmente para el almacenamiento y la reproducción digital de flujos sincronizados, además de permitir la entrega de datos a través de una amplia variedad de redes y protocolos.

Generalmente, los archivos ASF que contienen audio comprimido con el codec Windows Media Audio (WMA) usan la extensión .wma. Del mismo modo, los archivos ASF que incluyen contenido de audio, video o ambos, y que está comprimido con el codec Audio de Windows Media (WMA) o Video de Windows Media (WMV), usan la extensión .wmv. Por último, el contenido comprimido con cualquier otro codec utiliza la extensión .asf genérica.

Los objetivos principales del formato ASF son los siguientes:

- Soportar la reproducción eficiente desde servidores de streaming, servidores HTTP, y dispositivos de almacenamiento local.
- Permitir una sola composición multimedia que se presentará en una amplia gama de anchos de banda.
- Permitir el control de autoría de los flujos en servidores de streaming, especialmente en escenarios con ancho de banda limitado.
- Ser independiente de cualquier sistema de composición multimedia, sistema operativo, o protocolo de comunicaciones de datos.

Un archivo ASF puede contener múltiples flujos dependientes o independientes, incluyendo flujos para audio multicanal, o flujos de video con múltiples tasas de bits adecuados para la transmisión por diferentes anchos de banda. Estos flujos pueden estar en formato comprimido o sin comprimir, sin embargo, la mejor compresión se logra con los codecs Microsoft Windows Media Audio y Video.

El formato ASF apoya la distribución de contenido multimedia 'en vivo' y 'bajo demanda', que también puede ser utilizado como un vehículo para grabar o reproducir H.32X (por ejemplo, H.323 y H.324)

Windows Media Video

Es un conjunto de algoritmos de compresión, incluidos en el framework Windows Media e implementados en el set de tecnologías de video de Microsoft.

En la mayoría de los casos los archivos WMV son encapsulados en un contenedor ASF.

La primera versión del codec fue WMV 7, que se basa en la aplicación de estándareses de Microsoft MPEG-4 parte 2. Si bien todas las versiones de WMV soportan tasa de bits variable, promedio de velocidad de bits, y tasa de bits constante, la última vesión introduce varias características incluyendo soporte nativo para video entrelazado, pixeles no cuadrados e interpolación de marcos. Este codec es también el único que apoya video en la plataforma Microsoft Silverlight.

Microsoft afirma que su algoritmo WMV 9 establece una relación de compresión que es dos veces mejor que la de MPEG-4, y tres veces mejor que MPEG-2. También alega que en términos de eficiencia de compresión WMV 9 es de 15 a 50% mejor que WMV 8. Sin embargo, un informe sobre ensayos publicado por la EBU Technical Department en enero de 2005 demostró lo contrario.

Este informe concluye empíricamente que las nuevas generaciones de codecs no siempre son mejores a su versión anterior, WMV8 por ejemplo, funciona mejor que WMV 9 para 500 y 700 Kbps.

Common Image Format (352 píxeles/línea, 288 líneas por cuadro, 30 cuadros por segundo)

	250 Kbps	500Kbps	700 Kbps	1400 Kbps
WMV 8	52	55	62	73
WMV 9	50	43	50	70

Figura 1: Windows Media 9 vs. Windows Media 8.

Es difícil medir el rendimiento en la compresión de cierto estándar, dado que la calidad experimentada por un usuario depende de la calidad de cada aplicación codificadora o decodificadora y que partes del estándar estos aplican.

El consenso general indica que, aunque el MPEG-4 tiene la capacidad de proporcionar un ahorro en velocidad de bits en relación a versiones anteriores, otros diseños, como Windows Media Video 8 y el proyecto UIT-T H.26L han demostrado mejores soluciones a los avances de la ingeniería de codecs y la eficiencia asociada.

4. Librería WindowsMediaNET

El propósito de este proyecto es brindar a las aplicaciones desarrolladas en lenguajes .NET (C sharp, Visual Basic, entre otros) la posibilidad de utilizar las interfaces de Windows Media Format SDK y todos los métodos de waveIn, waveOut, mixer y MMIO que permiten el manejo de archivos multimedia en el sistema.

Esta librería, desarrollada por David Wohlferd y Bryce Kahle, ha sido testeada en C sharp y debería funcionar sin problemas en cualquier lenguaje .NET compatible. A partir de la versión 1.1 de la biblioteca se agrega el soporte para arquitecturas de 64bits.

4.1 Interfaces

En la mayoría de los casos, las llamadas a métodos de las interfaces son similares a las descritas en MSDN para el uso de Windows Media Format SDK.

Las principales diferencias entre ambas definiciones están relacionadas con los códigos de retorno y aquellas situaciones en donde se definen parámetros enteros (int), pero en realidad son enumerados.

Se cambian los nombres de las estructuras y enumerados para reflejar un uso más al estilo .NET.

A continuación de muesta una lista de los métodos incluidos en la biblioteca que han sido testeados.

WMCreateBackupRestorer
WMCreateEditor
WMCreateIndexer
WMCreateProfileManager
WMCreateReader
WMCreateSyncReader
WMCreateWriter
WMCreateWriterFileSink

WMCreateWriterNetworkSink
WMCreateWriterPushSink
WMIsAvailableOffline
WMIsContentProtected
WMCheckURLExtension
WMCheckURLScheme
WMValidateData

mixerClose
mixerGetControlDetails
mixerGetDevCaps
mixerGetID
mixerGetLineControls

mixerGetLineInfo
mixerGetNumDevs
mixerMessage
mixerOpen
mixerSetControlDetails

mmioAdvance
mmioAscend
mmioClose
mmioCreateChunk
mmioDescend
mmioFlush
mmioGetInfo
mmioOpen
mmioRead
mmioRename
mmioSeek
mmioSendMessage
mmioSetBuffer
mmioSetInfo
mmioWrite

waveInAddBuffer
waveInClose
waveInGetDevCaps
waveInGetErrorText
waveInGetNumDevs
waveInGetPosition
waveInMessage
waveInOpen
waveInPrepareHeader
waveInReset
waveInStart
waveInStop
waveInUnprepareHeader

waveOutBreakLoop
waveOutClose
waveOutGetDevCaps
waveOutGetErrorText
waveOutGetNumDevs
waveOutGetPitch
waveOutGetPlaybackRate
waveOutGetPosition
waveOutGetVolume
waveOutMessage
waveOutOpen
waveOutOpen
waveOutPause
waveOutPrepareHeader
waveOutReset
waveOutRestart
waveOutSetPitch
waveOutSetPlaybackRate
waveOutSetVolume
waveOutUnprepareHeader
waveOutWrite

La siguiente es una lista de las interfaces testeadas que expone la librería en su última versión.

INSNetSourceCreator
INSSBuffer
INSSBuffer2
INSSBuffer3
INSSBuffer4
IWMAddressAccess
IWMAddressAccess2
IWMBackupRestoreProps
IWMBandwidthSharing
IWMClientConnections
IWMClientConnections2
IWMCodecInfo
IWMCodecInfo2
IWMCodecInfo3
IWMCredentialCallback
IWMDRMEditor
IWMHeaderInfo
IWMHeaderInfo2
IWMHeaderInfo3
IWMIndexer
IWMIndexer2
IWMInputMediaProps
IWMIStreamProps
IWMLanguageList
IWMLicenseBackup
IWMLicenseRestore
IWMMediaProps
IWMMetadataEditor
IWMMetadataEditor2
IWMMutualExclusion
IWMMutualExclusion2
IWMOutputMediaProps
IWMPacketSize
IWMPacketSize2
IWMPlayerHook
IWMProfile
IWMProfile2
IWMProfile3
IWMProfileManager
IWMProfileManager2
IWMProfileManagerLanguage

IWMPropertyVault
IWMReader
IWMReaderAdvanced
IWMReaderAdvanced2
IWMReaderAdvanced3
IWMReaderAdvanced4
IWMReaderAdvanced5
IWMReaderAllocatorEx
IWMReaderCallback
IWMReaderCallbackAdvanced
IWMReaderNetworkConfig
IWMReaderNetworkConfig2
IWMReaderPlaylistBurn
IWMReaderStreamClock
IWMReaderTypeNegotiation
IWMRegisterCallback
IWMSInternalAdminNetSource
IWMSInternalAdminNetSource2
IWMSInternalAdminNetSource3
IWMStatusCallback
IWMStreamConfig
IWMStreamConfig2
IWMStreamConfig3
IWMStreamList
IWMStreamPrioritization
IWMSyncReader
IWMSyncReader2
IWMVideoMediaProps
IWMWriter
IWMWriterAdvanced
IWMWriterAdvanced2
IWMWriterAdvanced3
IWMWriterFileSink
IWMWriterFileSink2
IWMWriterFileSink3
IWMWriterNetworkSink
IWMWriterPostView
IWMWriterPostViewCallback
IWMWriterPreprocess
IWMWriterPushSink
IWMWriterSink

4.2 Perfiles

Existen varias maneras de configurar el objeto WMWriter (escritor de Windows Media), sea como un filtro DirectShow o un objeto WindowsMedia, la forma más sencilla de hacerlo es por medio de la utilización del IWMProfileManager:: LoadProfileByID. Este método tiene un GUID que se corresponde con distintas entradas en el archivo de definición WMSysPr9.prx de Windows. Estos perfiles se utilizan para especificar diversos parámetros de salida del video (por ejemplo la tasa de transferencia de datos, el tamaño del marco de video, entre otros).

Lamentablemente el archivo WMSysPr9.prx no contiene muchos perfiles definidos, hecho que restinge las posiblidades de configuración de archivos multimedia.

Para sobrellevar esto es posible utilizar métodos fuera de la interfaz IWMProfile y así crear conjuntos de flujos y valores multimedia apropiados. Probablemente la mejor manera de crear un perfil personalizado sea usar la herramienta GenProfile (desde el Windows Media Format SDK) y luego cargarlo mediante el uso del método IWMProfileManager:: LoadProfileByData.

Un punto fundamental a tener en cuenta si el perfil contiene dos flujos (audio y video), es la necesidad de conectar ambos flujos al objeto para que el archivo pueda ser escrito. Por lo tanto, si se utiliza un GUID al llamar LoadProfileByID por ejemplo, es necesario asegurarse que los flujos estén conectados.

5. Formato MPEG-4

Es una especificación de la Organización internacional de estándares (ISO, International Standards Organization) que abarca muchos aspectos de la presentación multimedia incluida la compresión, la creación y la entrega. La definición de contenedores de archivos y la compresión de video son dos entidades diferentes e independientes de la especificación

MPEG-4.

El formato de archivo MPEG-4, según se define en la especificación MPEG-4, contiene video codificado con MPEG-4 y audio codificado con Advanced Audio Coding (AAC). Suele utilizar la extensión .mp4. Es posible reproducir archivos multimedia .mp4 en Windows Media Player si instalan los paquetes decodificadores de MPEG-4 compatibles con DirectShow.

Microsoft ha decidido implementar la parte correspondiente a la compresión de video del estándar MPEG-4. creando los siguientes codecs de video basados en MPEG-4:

- Microsoft MPEG-4 v1
- Microsoft MPEG-4 v2
- Microsoft MPEG-4 v3
- ISO MPEG-4 v1

El contenido de video MPEG-4 se puede, también, codificar y almacenar en un contenedor de archivos ASF mediante herramientas de Windows Media.

6. Referencias

- EBU Technical Media Technology and Innovation – último acceso: 29/06/09
  http://tech.ebu.ch/
- Overview of the ASF Format – último acceso: 29/06/09
  http://msdn.microsoft.com/en-us/library/dd757562(VS.85).aspx
- Subjective Quality of Internet video codecs, EBU Technical Department, Janury 2005 – último acceso: 29/06/09
  http://www.ebu.ch/en/technical/trev/trev_301-samviq.pdf
- Windows Media Format SDK – último acceso: 29/06/09
  http://msdn.microsoft.com/es-es/library/dd757738(en-us,VS.85).aspx
- WindowsMedia.NET – último acceso: 29/06/09
  http://windowsmedianet.sourceforge.net/

4. Servidor de streaming

1. Introducción

En el presente anexo se describirán, en forma breve, los conceptos generales sobre streaming de contenidos en redes y sus distintos métodos de entrega.

Una vez comprendida la metodología, se presentan las principales características y configuraciones necesarias del producto Windows Media Services9 de Microsoft seleccionado para dar soporte a la emisión de entrevistas en vivo.

Finalmente, se explican detalles y parámetros a tener en cuenta para obtener el mejor rendimiento del sistema.

1.1 Descripción de streaming de video

El Streaming de video es un término aplicado a técnicas de compresión y buffering que permiten transmitir y ver video en tiempo real a través de Internet.

En lugar de descargar un archivo de video completo, la audiencia puede descargar pequeñas porciones del archivo digital desde Internet. Estos archivos de video pueden ser muy grandes, por lo que podría tomar un tiempo excesivo descargarlo completamente antes que el usuario final pueda reproducirlo.

En la técnica de streaming de video, sin embargo, el programa descarga el archivo en porciones formadas por paquetes más pequeños, que permiten a la aplicación cliente ir reproduciendo el video a medida que está siendo transferido. En otras palabras, el streaming de multimedia es la transferencia simultánea de medios digitales como archivos de audio, video o datos, a través de una aplicación servidor que se pueden visualizar en tiempo real. Una vez que este flujo se ha descargado y visto, el archivo físico no permanece en la computadora del usuario cliente. La operación es similar a una emisión de televisión donde el receptor presenta las imágenes y sonidos ante el público para que este pueda ver o escuchar. Una vez que el usuario cierra la aplicación del lado cliente, el archivo es automáticamente descartado y sólo se puede ver accediendo nuevamente al sitio de Internet que lo contiene.

El caso de emisión de video en vivo, funciona en forma similar, con la diferencia que la señal está siendo producida a medida que es consumida por la audiencia.

1.2 Tecnología detrás del streaming de video

Antes de la creación de un video digital a los efectos de distribuirlo, es necesario contar con la garantía de tener: (a) el video en el formato correcto, (b) acceso a un servidor de streaming, y (c) el ancho de banda suficiente para descargar y subir el archivo de video.

El hardware para la creación de materiales multimedia requerido es prácticamente el mismo que para la creación tradicional de video digital. Estos componentes de hardware y software incluyen una cámara de video (analógica o digital), tarjeta de captura de video, software de codificación de video, una computadora con espacio suficiente para almacenar los archivos generados y una conexión de red de alta velocidad.

Otros elementos deben ser considerados para obtener buenos resultados en el contenido generado. Estas consideraciones incluyen (a) buena base o trípode para eliminar el movimiento, esto ayuda a reducir el trabajo que el codificador realiza para compensar dicho movimiento, (b) micrófono en la dirección adecuada para ayudar a reducir el ruido de fondo y captar con mejor volumen de sonido y (c) una iluminación adecuada.

Es también importante asegurarse que el equipo sea lo suficientemente potente como para satisfacer el desarrollo de video digital. De acuerdo con el Joint Information Systems Committee en su artículo Click-and-Go Video Streaming guide, la computadora debe cumplir con las siguientes especificaciones para manipular adecuadamente el streaming de video digital:

- Procesador mínimo 1.2GHz o superior.
- 256MB de RAM mínimo (recomendado 512MB).
- Rápida unidad de HDD (recomendado 80GB 7200RPM o más).
- Tarjeta gráfica (mínimo 32MB de RAM).

La mayoría del streaming de video se presenta en uno de los siguientes formatos: (a) RealNetworks RealMedia (b) Microsoft Windows Media (ASF Avanced System Format), y (c) Apple Quicktime. Dichos formatos son propietarios y los contenidos producidos por cada uno pueden ser vistos mediante un reproductor que cada empresa ofrece en forma gratuita. Los tres soportan el protocolo de streaming en tiempo real (RTSP) para transmitir archivos en red.

Otros formatos para el streaming de video pueden ser MPEG-2 y MPEG-4. Este último es un estándar desarrollado específicamente para Web y entrega de video en tecnologías móviles.

Los pasos a seguir para crear y entregar los contenidos multimedia son relativamente simples. En primer lugar, se debe registrar dicho contenido por medio de un dispositivo que permita capturar secuencias de imágenes y sonido. El siguiente paso es digitalizar la captura obtenida, si el video había sido capturado con una cámara digital, este proceso no es necesario. En caso contrario tendrá que ser conectado a un dispositivo de captura de video y el software apropiado para digitalizar el archivo multimedia. Una tarjeta de captura de video traduce el archivo analógico en señales digitales que la computadora puede leer.

El video digital debe ser comprimido a través de un codificador que utiliza codecs (compresores/descompresores) de video que ayudan a reducir la cantidad de datos, eliminando repeticiones innecesarias. Al codificar los videos en un formato para streaming, se debe considerar la forma en que el video se comprime.

Existen tres criterios que se deben tomar en cuenta para conformar la calidad del streaming de video: (a) el *frame rate*, que designa el número de imágenes fijas que componen un segundo de video, (b) la profundidad de color o el número de bits que el equipo asigna a cada píxel, y (c) la resolución que es medida por el número de píxeles. Dichos parámetros pueden afectar en gran forma el tamaño y la calidad del archivo de video que será visto por la audiencia.

Una consideración final para la creación y emisión de videos es encontrar el servidor adecuado para contener y difundir dicha información.

La solución de *RealPlayer* es de distribución gratuita y está disponible para todas las plataformas.

El producto *RealNetworks Helix Server* está disponible en Windows, Linux, y sistemas operativos Unix soportando todos los formatos de archivos, incluyendo RealMedia, Windows Media, QuickTime, MP3, MPEG-4, entre otros.

El *Helix Basic Server*, de distribución gratuita, es limitado a un máximo de cinco clientes simultáneos, mientras que la versión paga de *Helix Universal Server* es un servidor completamente equipado que permite más de 100 clientes simultáneos.

Tanto el servidor, como el reproductor de *QuickTime* son gratuitos. El reproductor está disponible para la gran mayoría de las plataformas, mientras que el servidor sólo es provisto para MAC OS en forma nativa.

La solución *Windows Media Services9* de Microsoft y el reproductor Windows Media Player son distribuidos en forma gratuita, pero sólo están disponibles para versiones de Windows.

Según la Joint Information Systems Committee (2002) un servidor de streaming genérico exigiría las siguientes especificaciones:

- Procesador 1GHz o superior.
- 1GB de RAM o más.
- Disco duro buena capacidad de almacenamiento.

- Conectividad de red rápida.

Cuando el usuario cliente selecciona la emisión, el servidor comienza el proceso de streaming mediante el envío de pequeños paquetes de información a través de la red. Una vez recibidos suficientes datos, el reproductor comienza a mostrar la señal, mientras que al mismo tiempo actúa como un amortiguador (buffer) recogiendo los nuevos paquetes de datos que llegan desde el servidor. Si la velocidad de conexión no es buena, entonces se reduce bastante la performance del streaming. Así elegir la mejor opción disponible en términos de ancho de banda es sumamente importante porque si el servidor no tiene suficiente capacidad de transferencia para hacer frente a las demandas de los usuarios, se recibirán señales de mala calidad.

1.3 Limitaciones a considerar

El ancho de banda insuficiente para emisiones es un inconveniente que debe ser considerado a la hora de planificar la infraestructura a construir.

Dado que el flujo multimedia puede hacer uso intensivo del ancho de banda, los usuarios que disponen de módems 56K o incluso ADSL pueden tener dificultades al reproducir dicho flujo. Es así que la congestión de Internet puede provocar retrasos en la reproducción y mala experiencia a nivel de usuario final.

Además, la competencia de tecnologías para el control de normas es otro obstáculo para el streaming de video. Diferentes proveedores han establecido sus propias normas en materia de streaming de video, por lo que si un usuario no tiene el reproductor apropiado, no puede ver el contenido distribuido.

2. Protocolos de transmisión

2.1 Transmission Control Protocol (TCP)

El protocolo de transmisión TCP se considera un protocolo confiable en el sentido de que detecta errores y reenvía los paquetes que fueron dañados o perdidos en la red. Siendo así una aplicación podría identificar que recibió los paquetes 1, 2, 4 y 5, pero no el numero 3. Este protocolo también realiza una función de control de flujo, lo que hace que el remitente se retrase si el receptor que participa de la conexión no puede manejar la velocidad de envío.

Estas características hacen a TCP un protocolo orientado a conexión y de alta calidad, que asegura la obtención de todos los datos enviados por la red. No obstante, hay un precio que pagar por los beneficios de detección de errores, control de flujo y reenvío automático. Esto es, cuando el tráfico es importante y las conexiones están desbordadas causando la pérdida de paquetes, TCP envía los datos más lentamente teniendo que retransmitir una gran cantidad de paquetes.

2.2 User Datagram Protocol (UDP)

Como ya se expuso, TCP se considera un protocolo fiable para entregar datos desde un extremo a otro de la red, pero la fiabilidad no siempre es la clave. En algunos casos, es preferible un enfoque más ligero donde se pierda en confiabilidad pero se gane en eficiencia de entrega. El protocolo de datagramas de usuario (UDP) implementa un mecanismo de entrega que no agrega muchas características encima de IP. Los hosts envían datagramas que son básicamente paquetes con una dirección IP y puerto de host destino.

La función de checksum, que también incluye, es un modo de saber si el paquete fue recibido intacto o sufrió daños en la transmisión. En la figura se puede ver la comparación conceptual entre TCP y UDP.

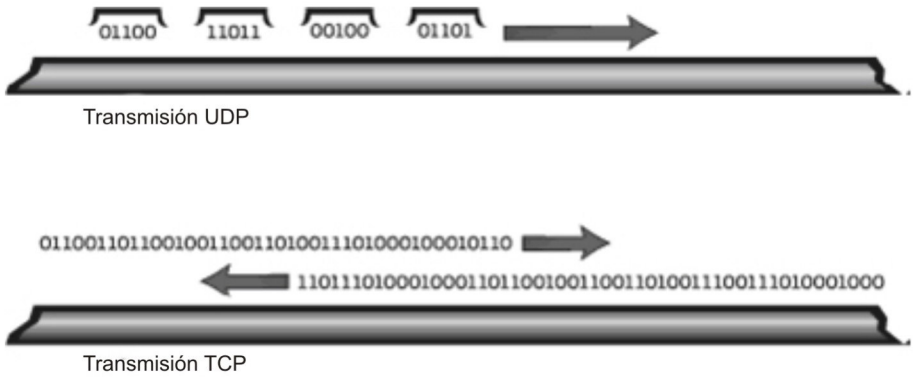


Figura 1: Comparación de protocolos TCP y UDP.

El protocolo UDP deja que la aplicación negocie el reenvío de un paquete perdido o dañado. A diferencia de TCP, UDP no asume que los paquetes perdidos deben ser reenviados automáticamente. Esto tiene ventajas obvias para el streaming de video, dado que en este escenario es preferible saltearse un paquete perdido antes de esperar a que llegue su reenvío. Por otro lado, UDP no tiene paquete de pedidos integrado, por lo que la aplicación también tiene que agregar el número de secuencia de los datagramas que manda. Las aplicaciones tienen que llevar su propia contabilidad para saber si un paquete ha sido rechazado.

Desde la perspectiva de programación, se puede comparar a UDP y TCP como una caja de cambios manual o automática respectivamente. Es más difícil manejar con una caja de cambios manual, pero de esta forma potencialmente se tiene mayor control sobre la velocidad, la aceleración y el ahorro del combustible. Siguiendo con la analogía, es más difícil escribir aplicaciones UDP porque es necesario programar explícitamente el control de flujo y la manipulación de paquetes. Los autos con cambios automáticos, en cambio, son más fáciles de manejar, no optimizan el uso de combustible, ni obtienen el mejor rendimiento del auto. Al igual que manejar un auto con transmisión automática, TCP es mucho más simple de programar, pero cuando las entregas se descontrolan, no hay mucho que hacer en pro de mejorar la situación.

El protocolo UDP cuenta con una serie de ventajas sobre TCP que lo hacen el preferido por programadores y diseñadores de redes para el streaming de video.

- Demora: si se pierde un paquete en UDP, el servidor puede seguir enviando paquetes al receptor. Si se pierde un solo paquete, eso significa solo uno o dos cuadros de video, por lo que el reproductor puede continuar. Cuando un paquete se pierde en TCP, el protocolo detiene el envio de datos y trata de recuperar el paquete de nuevo. Esto provoca un fenómeno llamado "jitter"que se traduce en tiempos desparejos y retrasos en los datos.
- Control de flujo: a diferencia de TCP, UDP no tiene un control de flujo. El control de flujo de TCP hace que se disminuya la velocidad de envío cuando el receptor no puede aceptar los datos. Sin embargo, por lo general el problema no es del receptor, si no de alguna condición temporal en la red. Por lo tanto solo UDP puede enviar datos a un ritmo constante, independiente de lo que haga la red.
- Baja sobrecarga: UDP es un protocolo sencillo que empaqueta, marca y envía los datos dejando el trabajo duro a la aplicación.

3. Protocolos de streaming

El protocolo HTTP, se basa en TCP y está optimizado para la recuperación de archivos. Sin embargo, no tiene ningún concepto de transferencia en tiempo real, es decir, los envíos toman el tiempo que sea necesario. En este tipo de protocolo, el cliente y el servidor se turnan para hablar dado que no es permitida la charla bidireccional.

Como se comentó anteriormente, UDP es el protocolo preferido para transferencia de streaming en tiempo real. Este protocolo puede enviar paquetes a un ritmo constante independientemente de la congestión de la red o la capacidad de la aplicación para recibirlos. Para modelar un protocolo de streaming multimedia basado en UDP se debe tener en cuenta la necesidad de realizar una serie de tareas.

- Configuración: Proporcionar los comandos start, stop, fast-forward, rewind y track skip.
- Transporte: Brindar una forma de entrega de múltiples flujos, detectando posibles paquetes faltantes.
- Sincronización: Proveer un método de sincronización entre flujos en tiempo real.
- Monitoreo de calidad: Brindar una manera de informar al servidor la situación en cuanto a la pérdida de paquetes y calidad de reproducción del cliente.

3.1 Protocolo de transporte en tiempo real (RTP)

La Internet Engineering Task Force (IETF) ha estandarizado un conjunto de protocolos para la entrega de video. El protocolo de transporte en tiempo real (RTP por sus iniciales en inglés), brinda todos los aspectos referentes al transporte y la sincronización a los que se hacen referencia en la sección anterior. El protocolo RTP normalmente se ejecuta sobre UDP, pero también puede funcionar a través de TCP, y puede ejecutarse sobre sistemas de transporte que no son de Internet. Se ocupa de la distribución de paquetes, aunque no asegura su entrega en tiempo real. Empaqueta los datos de audio y video con la suficiente información de tiempo como para que estos sean sincronizados por el receptor. Además, RTP es considerada la forma estándar de entregar multimedia sobre UDP en redes multicast.

Otra especificación de RTP es la RTCP (sigla en inglés para Protocolo de Control en Tiempo Real), que trabaja conjuntamente con RTP para proporcionar un canal de control útil para la vigilancia de la calidad. Los servidores mandan paquetes RTCP a todos los clientes y estos los devuelven periódicamente para informarle al servidor la calidad del stream que reciben. De esta manera, el servidor puede moderar la calidad de lo que envía.

A mediados del año 1996, el Protocolo de Streaming en Tiempo Real (RTSP), suministraba las características de configuración para la entrega de video en servidores de streaming. El protocolo se modeló usando HTTP como referencia y puede trabajar conjuntamente con RTP, donde se utiliza RTSP para establecer la conexión y RTP para entregar los datos.

3.2 Protocolo de Microsoft Media Server (MMS)

A fines de los '90, Microsoft creó su propio conjunto de protocolos para la entrega de multimedia. A pesar de que ya usaba RTP en su aplicación para conferencias *NetMeeting*, no habían implementado RTSP en ninguno de sus productos.

Fue así que Microsoft creó el protocolo MMS (Servidor Multimedia de Microsoft en inglés), integrando la mayoría de las características de RTP, RTCP y RTSP. Con el fin de llegar a la mayor audiencia posible, se diseñó el protocolo en varias versiones diferentes, cada una para un tipo más restringido de red:

- MMSU viaja sobre UDP para tener una entrega más eficiente.
- MMST transporta sobre TCP para redes que no permiten tráfico UDP.
- HTTP lleva el protocolo MMS sobre HTTP para redes que sólo permiten tráfico HTTP por los firewalls.

De esta forma se va descendiendo a protocolos menos restringidos hasta que el audio y video comienza a recibirse. Este enfoque se ilustra en la siguiente figura.

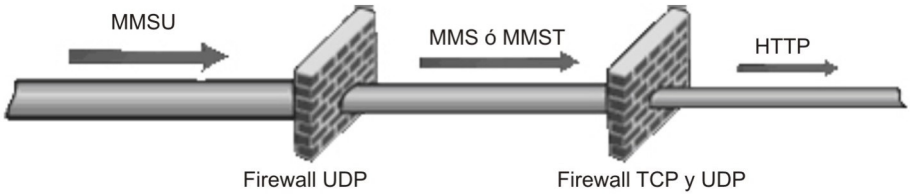


Figura 2: Descenso según restricciones de red.

Microsoft puso en uso este sistema de transporte de datos a fines de 2002 con la versión 9 del Windows Media, y forma parte de la tendencia a la estandarización del RTSP/RTP.

3.3 Protocolo Shoutcast/Icecast (ICY)

El protocolo de streaming Shoutcast/Icecast fue el comienzo para la emisión de radios MP3. La compañía Nullsoft, uso una versión levemente modificada del protocolo HTTP (llamado protocolo ICY), creando un servidor Shoutcast que podía mandar y recibir información en formato MP3 o casi cualquier codec de audio o video que pudiera fluir.

La primera versión de este protocolo consistía simplemente en MP3s metidos uno detrás de otro. En versiones posteriores se agregó la posibilidad de enviar también los nombres y títulos de los temas junto con las propias canciones, logrando que los reproductores mostraran dicha información. Finalmente se sumó el streaming de video.

La rápida aprobación y la estandarización de este protocolo se vió acompañada por una gran explosión de distintas aplicaciones capaces de reproducir estas emisiones, así como también una variedad de servicios. Cada reproductor que hay en el mercado funciona con streams de audio Shoutcast. Se puede decir entonces que, de alguna manera, este es el protocolo más interoperable y compatible que existe para el streaming de audio.

3.4 Streaming a través de firewalls

Los protocolos RTP y RTSP son estándares de streaming en Internet muy utilizados para el transporte de video en tiempo real que habitualmente funcionan sobre UDP. Sin embargo, por diferentes razones, UDP no siempre está disponible como opción de transporte. Una de las principales razones es que muchos de los firewalls corporativos están diseñados para bloquearlo, en el supuesto de que permitirlo promueva la inundación de la red con flujos que consuman gran ancho de banda. En general, UDP es comprensiblemente asociado con el uso intensivo del ancho de banda de las aplicaciones de Internet, tales como la telefonía por Internet, streaming de audio/video y video juegos. También es razonable que las empresas que buscan evitar que se navegue por Internet para cosas que no están vinculadas al trabajo, encuentren sencillo cortar todo el tráfico UDP en el firewall y bloquear así todos los servicios que dependen de él.

A fines de los '90, los principales proveedores de streaming tenían sistemas que corrían sobre UDP y podían trabajar también, aunque con problemas de rendimiento, sobre TCP. Las corporaciones comenzaban a restringir sus redes tanto que hasta los TCP generales no funcionaban, solo funcionaba el tráfico TCP en el puerto 80. Los firewalls están cada vez más restrictivos y ya casi no se puede usar TCP porque solo se permite el tráfico HTTP, como se muestra en la figura 3.

Como solución para esta situación se planteó un nuevo concepto: el protocolo de encapsulamiento. Indudablemente, HTTP no es un protocolo que permita una conexión permanente, pero tiene una característica por la cual dicha conexión puede ser mantenida abierta y en uso. Esta característica fue diseñada para situaciones en las que el usuario recibe muchos archivos de un mismo sitio Web.

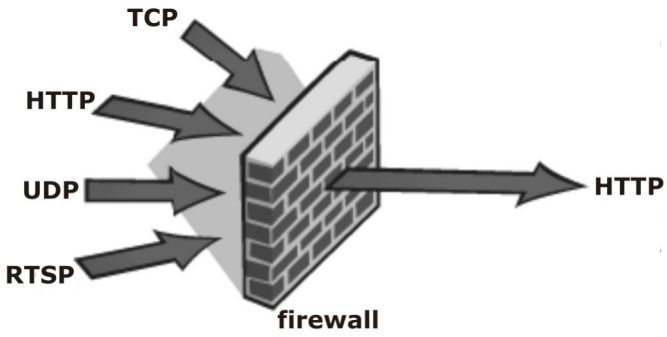


Figura 3: Firewalls bloquean todo tipo de tráfico, excepto el HTTP.

En respuesta a este entorno "solo-HTTP" los desarrolladores tuvieron que encontrar una forma de hacer ingresar video desde Internet en estas redes. Para esto, se desarrollaron protocolos de transporte a través de HTTP usando túneles. Con estos túneles, todos los paquetes que hubieran usado UDP como transporte se construyen normalmente, pero luego son enviados sobre conexiones HTTP. La figura ilustra cómo RTSP/RTP se pasa a través de un firewall encapsulado como tráfico HTTP.

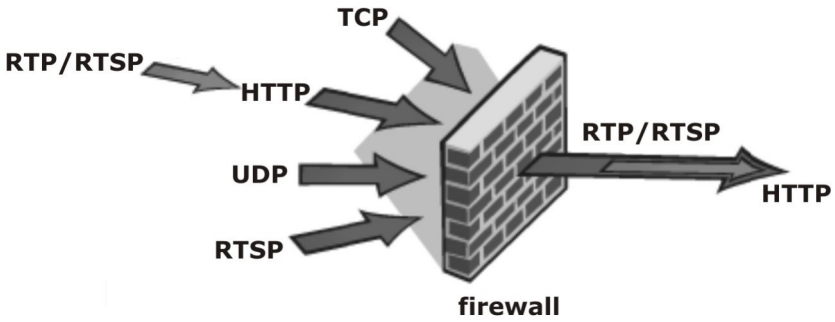


Figura 4: Se encapsula streaming en HTTP.

En esencia, lo que el servidor de streaming busca es enviar páginas web grandes para engañar al firewall y que de esta forma, el video pueda atravesar la barrera impuesta. Este método es bastante ineficiente, pero muy útil en ambientes restringidos.

4. Windows Media Services9 Series

4.1 Introducción

Windows Media Services (WMS) es la familia de software diseñado y desarrollado por Microsoft para brindar la mejor experiencia posible en streaming de multimedia para plataformas Windows.

Es una tecnología que permite distribuir contenidos de audio y video a clientes conectados en una intranet o Internet. Estos clientes pueden ser aplicaciones personalizadas y dispositivos especiales que reproducen el contenido, o incluso equipos que ejecutan servicios de Windows Media usados para resdistribuirlo.

El contenido que el servidor de streaming Windows Media distribuye a los clientes puede ser un flujo en vivo o archivos multimedia pregrabados.

La solución cuenta con herramientas que permiten una fácil configuración y administración de uno o más servidores Windows Media que ofrezcan su contenido en la red, incluso aunque no se esté familiarizado con los conceptos básicos de streaming.

4.2 Streaming

Para ofrecer los distintos contenidos por medio de un método de streaming, se debe almacenar dicho contenido generado en un servidor Windows Media y asignarle a este un punto de publicación. El caso de una emisión de flujo en vivo se resuelve en forma similar, con la salvedad de que se va inyectando en el punto de publicación a medida que se genera el contenido.

La audiencia que recibe la transmisión debe referenciar a una dirección URL que contiene el punto de publicación expuesto y a través de ella es capaz de reproducir el contenido multimedia.

La técnica de streaming utiliza el ancho de banda de manera más eficiente que la simple descarga de archivos, dado que el envío los datos a través de la red se realizan a la velocidad necesaria. Esta solución evita la saturación de la red y ayuda a mantener la fiabilidad del sistema. Normalmente existe un retraso entre el momento en que el flujo es recibido y el comienzo de su reproducción, dado que el cliente debe llenar un buffer de datos para cubrir retrasos o deficiencias en el flujo. Para poder transmitir contenido en tiempo real sin inconvenientes, la tasa de transferencia de bits debe ser menor que el ancho de banda de la red.

El impacto de la poca disposición de ancho de banda en el extremo cliente puede ser minimizado mediante el uso de varios bit-rate (MBR) en el contenido. De esta forma es posible cambiar a una tasa más baja de transferencia y evitar efectos indeseados.

4.3 Configuración de servicio Windows Media

Una vez que el contenido es generado, se debe crear, agregar y configurar un punto de publicación en el servidor de streaming a través del cual se deja el flujo disponible para su consumo. Dicho contenido multimedia se almacena en una ruta física del servidor Windows Media y es mapeado al punto de publicación. Cada punto puede ser definido según su característica y clasificado en dos tipos: demanda o difusión. Un punto de difusión es la mejor opción si se trata de transmitir contenido en vivo.

También es posible modificar la configuración de WMS para limitar el número de conexiones de clientes, crear medidas de seguridad para proteger el contenido, registrar la actividad de los clientes, entre otras.

Un punto importante a tener en cuenta en el momento de diseñar la estrategia de entrega es definir la forma en que se desea difundir el flujo de datos. A modo de aprovechar al máximo las capacidades de red disponibles, es posible (en algunas versiones) entregar el contenido como flujo *multicast* o hacerlo *unicast* si esta fuese la única opción.

Entrega de contenidos unicast

Unicast es una conexión uno a uno entre servidor y cliente, lo que significa que cada cliente recibe un stream y sólo aquellos que lo soliciten lo recibirán y podrán reproducirlo. Este método de streaming es el configurado por defecto en el servidor WMS para ofrecer contenido.

La siguiente figura muestra un escenario donde se entrega contenido por *unicast* mediante el acceso al punto de publicación TV1.

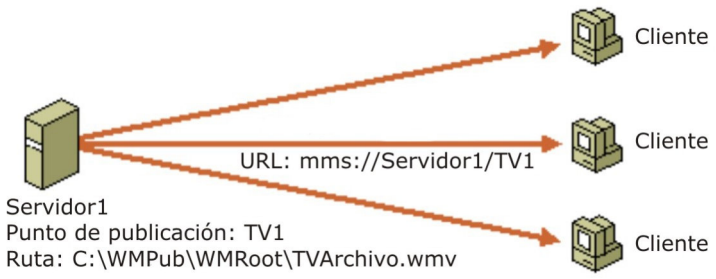


Figura 5: Emisión de flujo por unicast.

Entrega de contenido por multicast

El método multicast es una relación de comunicación uno a muchos entre un servidor Windows Media y los clientes que reciben el stream. De esta forma, todos los clientes reciben el mismo flujo de datos dado que existe un solo stream desde el servidor. Un flujo multicast consume la misma cantidad de ancho de banda que una sola conexión unicast. Por su característica, la entrega multicast puede ser útil en casos que el ancho de banda de redes de área local (LAN) es limitado, pero normalmente no puede ser usado en Internet.

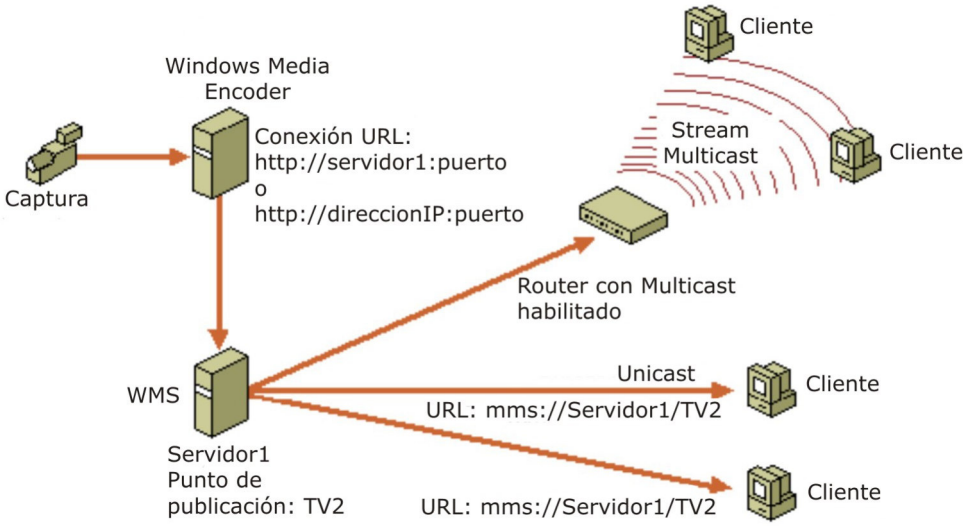


Figura 6: Esquema de distribución por multicast/unicast.

4.4 Características

El servicio de streaming de Windows Media Services fue diseñado pensando en lograr un servidor flexible, unificar componentes básicos y simplificar el proceso de administración.

Entre otras se pueden destacar las siguientes funcionalidades de WMS9 Series:

- MMS rollover: Los clientes que soportan el protocolo RTSP que intentan conectarse a un servidor WMS9 utilizando una URL que comienza con MMS, lo hacen por medio del protocolo rollover para obtener una óptima experiencia de streaming. Este protocolo permite pasar de MMS a MMS sobre UDP (mmsu) o TCP (mmst), o incluso HTTP, logrando que el cliente y el servidor negocien el mejor protocolo de streaming posible.
- Plug-ins: Permite varias formas de personalizar y configurar el servidor a través de una arquitectura basada en plug-ins.

- Puntos de publicación: El uso de puntos de publicación brinda una forma sencilla de presentar el contenido multimedia a los clientes.
- Server-Side playlists: Todos los puntos de publicación pueden distribuir contenido desde una lista de reproducción (playlist). Cada ítem en la playlist tiene un conjunto de atributos independientes que se pueden ajustar a cada necesidad. Es posible modificar las playlists aún cuando están siendo difundidas, lo que permite responder a ciertas situaciones sin tener que interrumpir la emisión.
- Streaming rápido: Esta facilidad permite una combinación de streaming y caching para proveer una mejor experiencia al usuario.
- Fecha/hora absoluta en playlist: Se agrega a los server-side playlist el atributo wallclock, el cual puede ser usado para automatizar el agendado de distribuciones mediante la asignación de un valor de fecha/hora expresado en UTC.

Un servidor corriendo Windows Media Services9 Series solo puede distribuir los siguientes formatos de archivo multimedia:

- Windows Media Audio (WMA).
- Windows Media Video (WMV).
- Advanced System Format (ASF).
- MP3.
- Windows Media playlist (WSX).

El servicio WMS no puede usar la capacidad de streaming inteligente para distribuir archivos MP3, pero es posible agregar imágenes codificadas en JPEG para intercalar avisos y publicidad en las playlist para distribuciones unicast.

Windows Media Protocol Rollover

A partir de Windows Media Player9 y Windows Media Services9, Microsoft añadió soporte para el protocolo RTSP. En ambos casos y con el fin de tener compatibilidad hacia atrás, tanto el reproductor, como el servidor mantuvieron su antiguo protocolo MMS.

Cuando el reproductor se conecta al servidor trata de usar el protocolo más eficiente para realizar la tarea. Debido a la presencia de firewalls no siempre es posible conectarse con el primer protocolo con el que se intenta.

Los tres posibles protocolos disponibles en el servicio WMS9 son MMS, RTSP y HTTP. En general es preferido el protocolo de transporte UDP, sin embargo, el reproductor primero debe establecer una conexión TCP con el servidor para configurar la sesión y negociar el stream UDP. Si la conexión no es posible, entonces el reproductor elegirá el siguiente protocolo disponible.

Moniker MMS

Es importante distinguir la diferencia entre el moniker y el protocolo MMS, dado que mientras el primero todavía es utilizado para que los reproductores puedan acceder al rollover antes descrito, el protocolo está en desuso.

Especificación explícita de protocolo

En las herramientas de reproducción de Windows también es posible especificar explícitamente el protocolo que desea utilizar cuando se conecta a un servidor. En lugar de utilizar el moniker MMS, se puede usar MMSU (MMS sobre UDP), MMST (MMS a través de TCP), RTSPU (RTSP sobre UDP), RTSPT (RTSP a través de TCP) o HTTP.

De esta forma, el reproductor queda vinculado al protocolo definido y no hace el rollover. Establecer explícitamente el protocolo debería ser utilizado solo en casos de prueba, ya que en producción es preferible usar el protocolo de rollover para una mejor experiencia.

4.5 Instalación

Es recomendable que la instalación del servidor de Windows Media se realice sobre la versión *Enterprise* del sistema *Windows Server 2003*. Se aconseja esta versión por su capacidad de manejar envíos con multicast.

Administración de contenidos

Requisitos mínimos:

- Procesador 133MHz
- 128MB de RAM
- 1.2GB de espacio libre en disco duro

Requisitos recomendados:

- Procesador 733MHz o superior
- 256MB – 64GB de RAM
- Soporte hasta 8 procesadores simultáneos
- Espacio libre en disco duro

Una vez instalado el sistema Windows Server 2003 Enterprise Edition, aparece un asistente que permite definir el tipo de servidor a crear. Dado que se intenta realizar streaming multimedia, se elije la opción *Agregar o quitar función* en la pestaña *Administrar las funciones de su servidor* donde se debe indicar que el servidor sea un *servidor de multimedia de transmisión por secuencias*. A continuación se siguen los pasos del asistente de configuración.



Figura 7: Asistente de creación.

Luego de configurada la dirección IP donde se encuentra el servidor, se crea un acceso en el panel izquierdo que apunta a éste, que luego permite administrar las funciones del servidor definido.

Una vez que se tiene el servidor de streaming instalado, se deben realizar ajustes a su configuración para personalizarlo de acuerdo a las condiciones y particularidades del entorno de red disponible y las consideraciones de seguridad básicas. Para definir la forma en que ofrecerá los contenidos a la audiencia, es necesario elegir *Protocolos de control* en la pestaña *Propiedades* y habliltar el que corresponda.

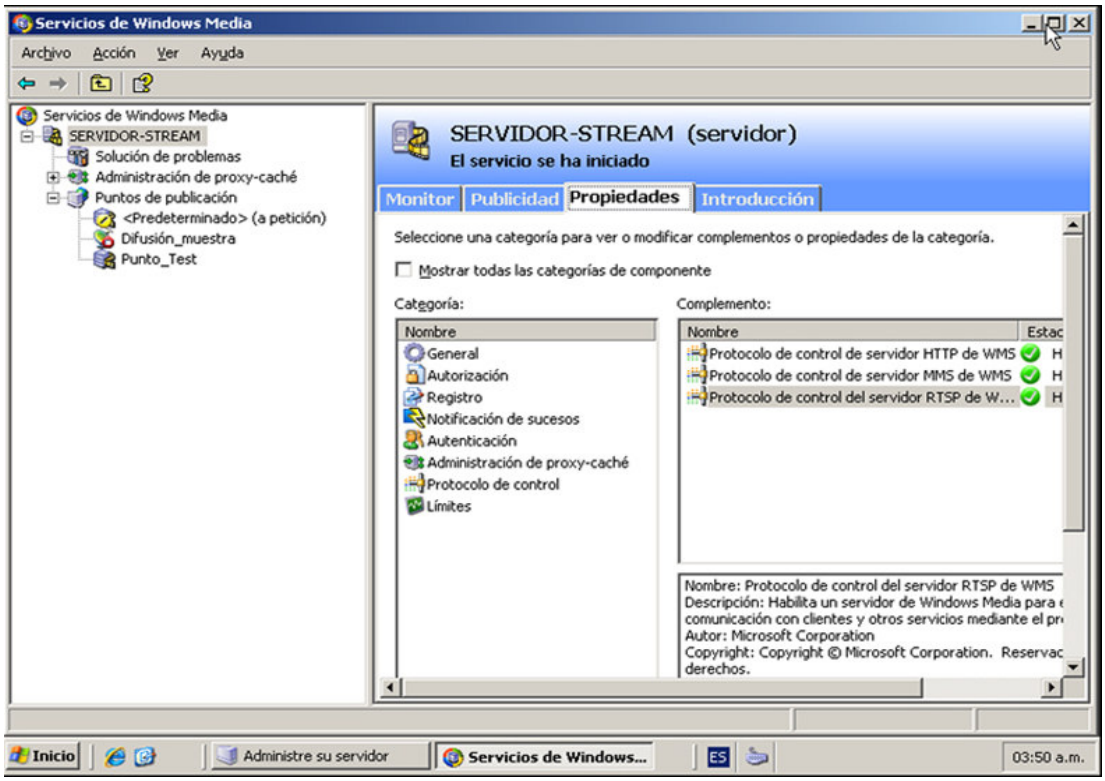


Figura 8: Configuración de protocolos de entrega.

Para comenzar a comunicarse con el servidor, también es necesario activar la autorización ACL de los puntos de publicación con permisos de escritura y creación para el usuario emisor definido. Para activar y configurar esta opción primero se elije *Autorización ACL de puntos de publicación de WMS* en *Autorización*.

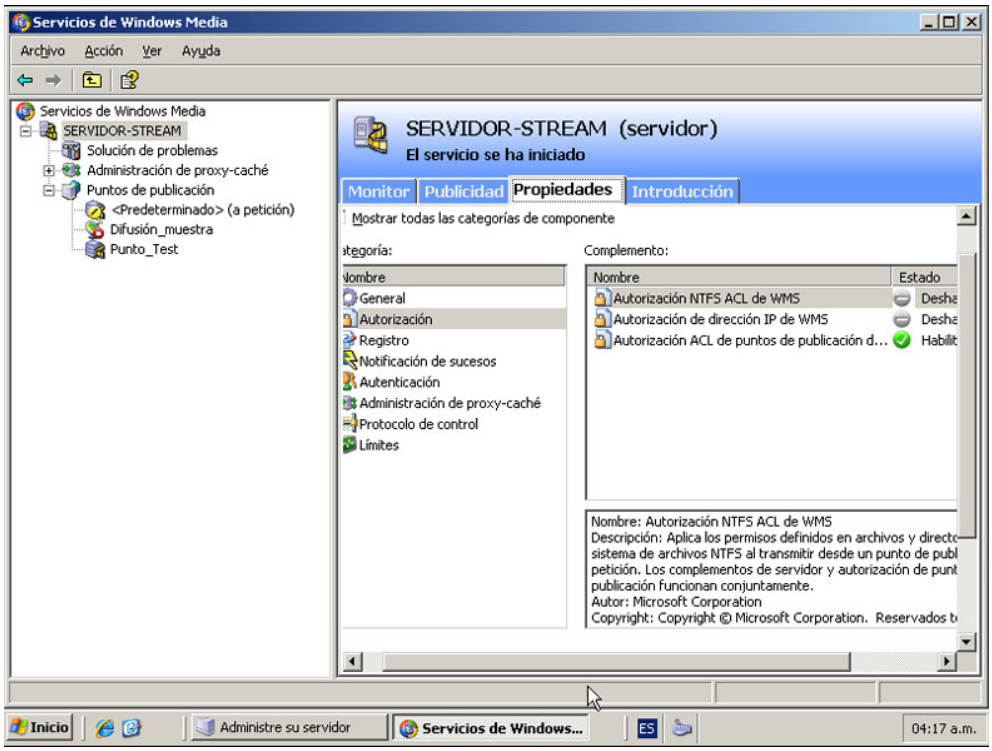


Figura 9: Configuración de autorización ACL.

Una vez en este punto, se ingresa a la opción de *Propiedades* para este item y se otorgan los permisos de escritura y creación al usuario genérico del sistema configurado para inyectar el flujo multimedia desde el codificador.

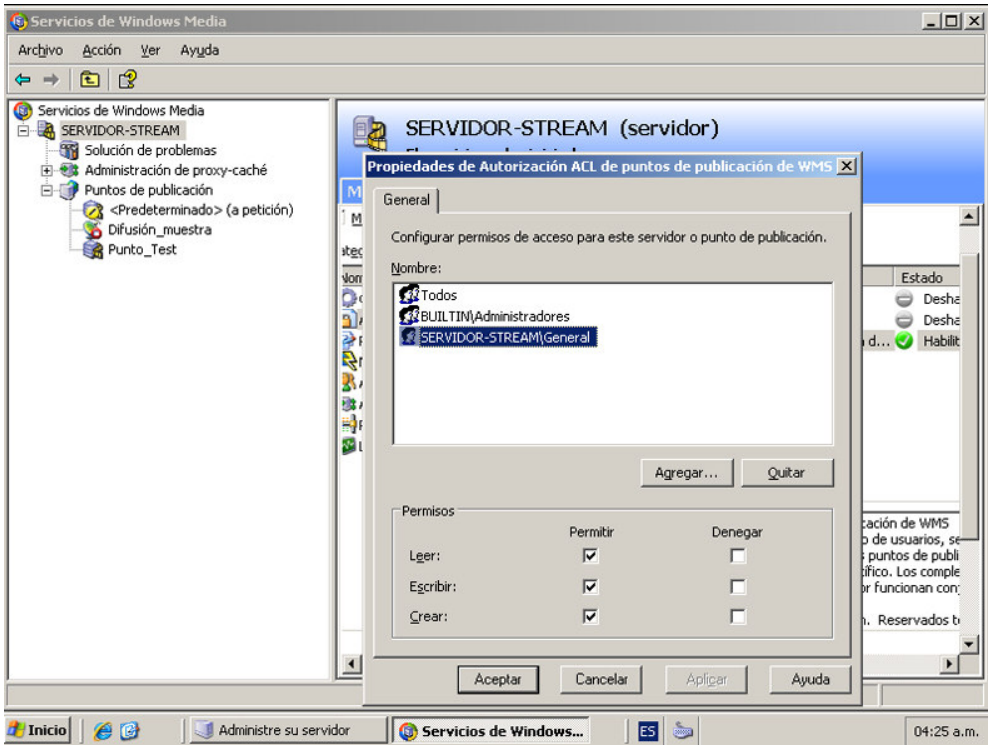


Figura 10: Asignación de permisos al usuario genérico.

Una vez que se han realizado todos estos pasos previos se cuenta con un servidor de streaming listo para recibir y enviar de contenido multimedia.

4.6 Rendimiento

La mayoría de los programas de computación utilizan áreas de almacenamiento temporal en memoria llamadas buffers. Típicamente, un programa realiza buffering de datos antes de mandarlos a un dispositivo, como un disco duro o impresora. Normalmente, los componentes Windows Media bufferean pequeños pedazos de los datos multimedia, por lo que estos pueden ser procesados a medida que se producen o envían a través de una red.

Reducción de demora en la transmisión

En un escenario de emisión, los buffers tienen un papel importante al garantizar la calidad de la transmisión ya que tanto computadoras, como dispositivos de red deben comprimir, codificar, distribuir, descomprimir e interpretar mucha cantidad de datos en tiempo real de manera muy rápida. Sin embargo, existe una gran desventaja en el uso de buffers grandes en una emisión en vivo; los buffers causan retrasos o latencias en la transmisión.

Una demora en la emisión es la diferencia entre el momento en que el audio y video en vivo son codificados y el momento en que estos son reproducidos. Esta demora es ocasionada principalmente por los buffers que almacenan los datos multimedia. Por ejemplo, si los buffers almacenan un total de 10 segundos de datos, el reproductor mostrará el evento 10 segundos tarde. Habitualmente, un retraso de hasta 20 segundos no representa un problema. Sin embargo, para cuando el tiempo es importante los componentes de Windows Media proporcionan una serie de maneras en las que se puede minimizar la demora sin causar una pérdida importante de imagen y calidad de sonido.

Buffer de un streaming multimedia

Generalmente cuando hay que describir el tamaño de un buffer, se suele pensar en términos de espacio de almacenamiento. Por ejemplo, un programa de procesamiento de textos podría utilizar 50KB para almacenar temporalmente un documento. Asimismo, cuando se describe un buffer de Windows Media, se suele pensar en términos de tiempo,

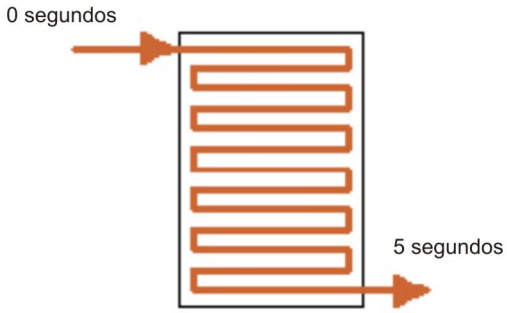


Figura 11: Buffer de tiempo.

Los datos entran a la parte superior del buffer en el segundo cero y salen cinco segundos después. Durante esos cinco segundos que los datos están en el buffer, un programa o algoritmo puede cambiar, copiar, borrar, reacomodar o guardar estos datos.

- **Corrección de errores**

Para que audio y video puedan ser reproducidos apropiadamente, la información multimedia debe ser recibida con un flujo continuo. Internet es un ambiente desafiante en ese sentido, ya que por su enorme tamaño, la cantidad de datos que deben ser transmitidos, su ancho de banda impredecible y las grandes distancias que debe recorrer la información, hacen que el entorno de trabajo sea menos favorable. A pesar de que todas estas condiciones son mejoradas día a día, es imprescindible contar con un sistema de corrección de errores para poder transmitir audio y video.

Windows Media está diseñado para distribuir stream con la mejor calidad posible a todos los usuarios, independientemente de las redes a las que estos pertenezcan o la velocidad

de conexión de la que dispongan. Para mitigar los errores causados por datos que se pierden, están dañados o incompletos, Windows Media Services y Windows Media Player usan buffers. Los datos en un streaming son enviados en paquetes numerados que incluyen la dirección de quien envía y la de quien recibe. El reproductor usa un buffer para acomodar los paquetes que recibe acorde con la numeración que traen. Si un paquete se daña, o no llega a tiempo, el reproductor lo pide nuevamente al servidor y éste pude reenviarlo desde el buffer.

Aunque la emisión se corte, la aplicación cliente podrá continuar con la reproducción dado que utiliza información almacenada en su buffer. Una vez restablecida la recepción de datos, estos vuelven a entrar y la presentación continúa normalmente a pesar que algunos paquetes hayan llegado en desorden.

- **Minimización de la demora**

Los buffers son parte esencial de la creación, envío e interpretación de los contenidos que le permiten a los componentes de Windows Media ofrecer una experiencia de alta calidad al usuario final. Sin embargo, cuanto más grande es el tamaño del buffer, más larga la demora de la emisión. Los valores de tiempo por defecto para los buffers definidos en los componentes de Windows Media equilibran calidad y demora en la mayor parte de las situaciones. Existen tres configuraciones que pueden modificarse para minimizar la demora de la emisión.

La siguiente ilustración muestra los buffers en cada uno de los actores involucrados en la emisión y cómo aportan a la demora.

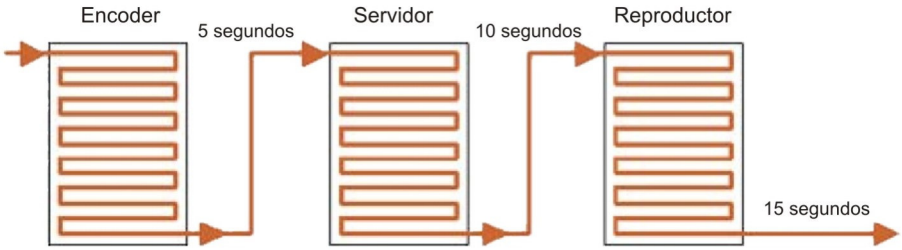


Figura 12: Composición de retardo por buffers.

El encoder almacena contenido para permitirle al codec optimizar la compresión. El servidor almacena los datos para permitir un inicio rápido; y el buffer del reproductor le ofrece al usuario una reproducción fluida. Asumiendo que cada buffer almacena cinco segundos de información, y tomando en cuenta las demoras que podrían agregarse por equipos presentes en la red, la demora total de la emisión podría ser de 15 a 20 segundos. No es posible eliminar completamente los buffers; de hecho los componentes de Windows Media no podrían funcionar sin ellos. No obstante, mediante la minimización de sus tamaños es posible reducir la demora en la emisión a tiempos que oscilan entre seis a nueve segundos, dependiendo de las condiciones de la red.

Cuando se reduce el buffer en el codificador (encoder) es necesario asegurarse que la calidad de imagen sea aceptable. El codec reduce los datos de movimiento, por lo cual es posible que se pierda la suavidad en el video. Si el contenido a emitir no incluye escenas de movimiento rápido, quizás ni siquiera se perciba degradación de la calidad.

Al reducir el tamaño de buffer del reproductor, es probable que la reproducción se vea continuamente interrumpida por el efecto buffering, lo cual sucede porque la aplicación se queda sin información. Es posible que el buffering no sea un problema para usuarios con un acceso a Internet estable y de alta velocidad. Sin embargo, aquellos con conexiones lentas podrían tener que aumentar el tamaño de su buffer. No hay que olvidar que se intenta equilibrar la calidad con la demora de la emisión, y que al reducir la demora es posible que comprometa otros aspectos.

Otra opción para bajar la demora en la emisión de contenido, es desactivar el buffer de inicio rápido (Fast Start) en el servidor de streaming Windows Media Services. Para desactivar

el uso de este buffer de debe modificar el punto de publicación de emisión creado por defecto ejecutando los siguientes pasos:

1- Hacer click en el punto de publicación de emisión y luego en el panel de detalles de la derecha, elegiendo *Propiedades*.

2- Hacer click en *Activar caché rápido* y luego marcar *Deshabilitar el almacenamiento en buffer*.

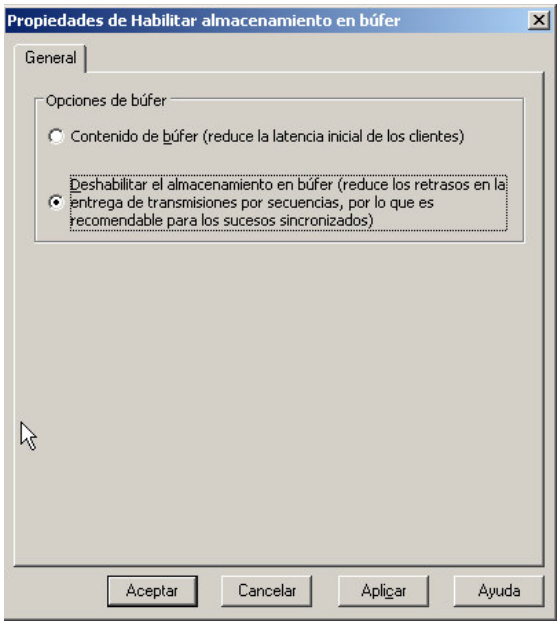


Figura 13: Desactivar el buffer de WMS.

4.7 Planificación de la capacidad de red

Si los clientes tienen un buffering excesivo, una razón podría ser que el servidor de Windows Media estuviese sirviendo a demasiadas conexiones simultáneamente. Debido a limitaciones de hardware, un servidor de Windows Media sólo puede transmitir una cantidad limitada de streams al mismo tiempo. Los servidores sobrecargados generalmente pierden información e interrumpen las transmisiones.

De manera alternativa, el servidor puede estar excediendo la capacidad de ancho de banda de la red. Es posible que la red tenga un punto débil o una falla, o incluso puede que no haya sido diseñada para transferir la cantidad de información que sus clientes requieren. Es posible implementar un conjunto de soluciones que permitan aliviar la carga de transferencia de datos del servidor o la red.

- **Fijar límites en el servidor**

Se puede configurar el servidor para que limite la cantidad de conexiones de clientes y el ancho de banda utilizado para no exceder las capacidades del servidor y de la red.

- **Crear un cluster de servidores**

Es posible utilizar un cluster para que un grupo de servidores de Windows Media trabajen juntos enviando el contenido. A pesar de que los clientes se conectan al cluster usando una única URL, todos los servidores comparten la carga de forma de reducirla sobre cada servidor individual.

- **Agregar servidores de distribución**

Se puede diseminar la carga de streaming sobre la red mediante la utilización de servidores de distribución en puntos estratégicos donde las demandas tienden a ser mayores. Esto puede mejorar sustancialmente el desempeño del streaming porque se reduce la distancia entre servidor y cliente.

Es posible estimar la capacidad de red necesaria utilizando la ecuación:

Capacidad de red requerida = *tasa de transferencia* x *volumen de audiencia*

Para estimar la tasa de transferencia promedio, se divide el tamaño del archivo que se está enviando por el tiempo de reproducción en segundos. Por ejemplo, un archivo digital multimedia de 2MB representa alrededor de 16.000.000 bits. Si el contenido tiene una duración de cerca de 1,5 minutos, el contenido enviado tiene una velocidad promedio de 180Kbps.

Para estimar el volumen de audiencia, es necesario determinar la cantidad más alta de usuarios simultáneos durante un evento de streaming.

Por último, un detalle a tener en cuenta es que la cantidad máxima de usuarios simultáneos depende de la configuración del servidor y del método de distribución que se utilice (unicast o multicast). Como regla general, un servidor Windows Media con 256MB de RAM puede servir hasta 1.000 streams unicast de 28,8Kbps. Dado que el streaming multicast requiere un único flujo desde el servidor Windows Media, en este caso un número ilimitado de usuarios puede conectarse simultáneamente al stream.

5. Referencias

- AboutVideoEditing. (2003b). How to shoot streaming videos for the web. Retrieved November 4, 2005 – último acceso: 29/06/09 http://www.aboutvideoediting.com/articles/shooting-streaming-video.shtml
- AboutVideoEditing. (2003a). Streaming video over the web. Retrieved November 4, 2005 – último acceso: 29/06/09 http://www.aboutvideoediting.com/articles/webstreaming-video.shtml
- Internet Engineering Task Force (IETF) – último acceso: 29/06/09 http://www.ietf.org
- Joint Information Systems Committee – último acceso: 29/06/09 http://www.jisc.ac.uk
- Stolarz, Damien; Mastering Internet Video: A Guide to Streaming and On-Demand Video, Addison Wesley, 2005.
- Update the Windows Media Server Platform – último acceso: 29/06/09 http://www.microsoft.com/windows/windowsmedia/howto/articles/UpgradeWMS9S.aspx
- Windows Media Services 9 Series SDK – último acceso: 29/06/09 http://msdn.microsoft.com/en-us/library/ms738748(VS.85).aspx

A5 – Anexos referidos a los resultados

1. Experiencia en WikiMed

Los resultados de la puesta en producción de *WikiMed* no se hicieron esperar. A pocos días de su instalación fue utilizada como apoyo educativo en el marco del curso 'El médico general y el paciente con cáncer' organizado por *EviMed* y la Comisión Honoraria de Lucha Contra el Cáncer. Su desarrollo durante los meses de junio y julio de 2008, fue el primer acercamiento a *WikiMed* de los médicos participantes del curso.

La propuesta de los organizadores fue la creación de un artículo generado por los propios médicos que tuvo como consigna: 'Estrategias para mejorar la comunicación entre los integrantes del equipo de salud, ante el paciente con cáncer'. En respuesta a la propuesta se obtuvo un artículo escrito con la participación de nueve médicos de distintos departamentos del país; Paysandú, Montevideo, San José, Soriano, Colonia y Rivera.

La segunda experiencia con *WikiMed* se realizó del 3 de octubre al 17 de noviembre de 2008 en el marco del curso 'Cesación de tabaquismo'. En esta oportunidad la propuesta de uso fue similar al anterior curso y la nueva consigna fue 'Cómo organizar una Unidad de Tabaquismo'.

Algunos datos estadísticos obtenidos de esta experiencia fueron los siguientes:

- 186 inscriptos al curso
- 19 participantes activos en la Wiki (10%)
- 2 editoras (médica especialista + comunicadora educativa)
- 3.172 palabras como resultado final

El artículo que se concluyó en esta oportunidad contó con la participación de médicos radicados en Montevideo, Paysandú, Maldonado, Colonia, Artigas, Treinta y Tres, Salto y Canelones.

Como conclusión se puede observar que de una experiencia a otra creció tanto la participación como el número de departamentos en el que se encuentran los participantes. Las distancias cubiertas indican la interacción de especialistas en departamentos tan alejados como Montevideo y Artigas o Colonia y Rivera.

En las siguientes imágenes se pueden ver comentarios asociados a los artículos realizados por los usuarios registrados, mostrando la aceptación por parte de los médicos y el uso de la Wiki para interactuar entre ellos.



¡Hasta luego! Elisa (Wiki-asistente)

hola [editar]

hola soy NOEMY MAINO

```
ES MI PRIMERA EXPERIENCIA, SALUDOS.
```

SALUDAR [editar]

Hola a todos en especial a las chicas el Aeropuerto

```
con quienes comparti el Taller en Monetevideo.
     Mi nombre es Noemy Maino de Tarariras Colonia
   Le envio un saludo a Elisa gracias por su paciencia, me fue e gran utilidad.
   Al Dr Lico un saludo le conozco por la paciente Sixta TORRES que te adora.
Por hoy los dejo.
```

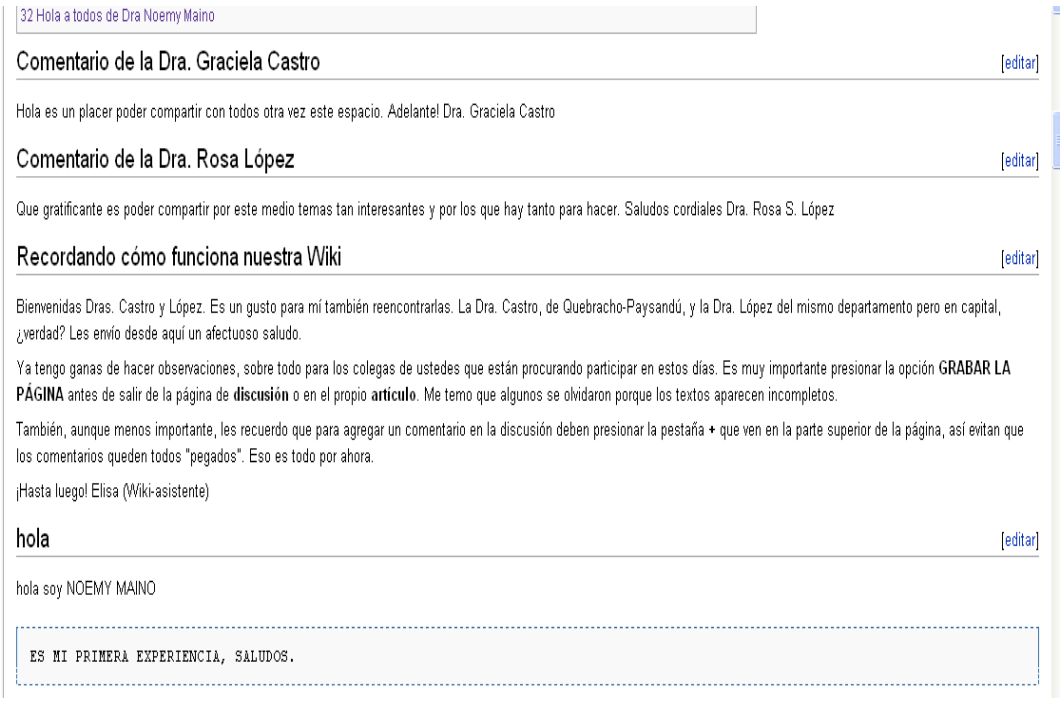
Comentario de la Lic. Araújo [editar]

Hola: primer curso on line para mi. Un gusto estar en esta discusión. Agregué algunas cosas respecto a cómo organizar una unidad de cesación de tabaquismo. Me pareció importante que no estuviera sólo la introducción los obstáculos y las propuestas, así que puse estrategias y organización como ítems apartes. Lo que estaba escrito antes no lo modifiqué, lo integré a mi propuesta.

Perdón, la que escribió anteriormente es Olga Araújo, Lic en Enfermería. Disculpen que no me presenté, porque pensé que mi nombre saldría automaticamente.

Edición del artículo [editar]

Figura 1: Comentarios en WikiMed, primera experiencia.



32 Hola a todos de Dra Noemy Maino

Comentario de la Dra. Graciela Castro [editar]

Hola es un placer poder compartir con todos otra vez este espacio. Adelante! Dra. Graciela Castro

Comentario de la Dra. Rosa López [editar]

Que gratificante es poder compartir por este medio temas tan interesantes y por los que hay tanto para hacer. Saludos cordiales Dra. Rosa S. López

Recordando cómo funciona nuestra Wiki [editar]

Bienvenidas Dras. Castro y López. Es un gusto para mí también reencontrarlas. La Dra. Castro, de Quebracho-Paysandú, y la Dra. López del mismo departamento pero en capital, ¿verdad? Les envío desde aquí un afectuoso saludo.

Ya tengo ganas de hacer observaciones, sobre todo para los colegas de ustedes que están procurando participar en estos días. Es muy importante presionar la opción **GRABAR LA PÁGINA** antes de salir de la página de **discusión** o en el propio **artículo**. Me temo que algunos se olvidaron porque los textos aparecen incompletos.

También, aunque menos importante, les recuerdo que para agregar un comentario en la discusión deben presionar la pestaña **+** que ven en la parte superior de la página, así evitan que los comentarios queden todos "pegados". Eso es todo por ahora.

¡Hasta luego! Elisa (Wiki-asistente)

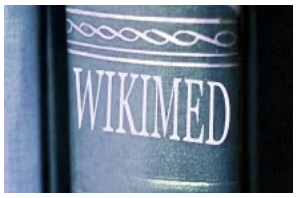
hola [editar]

hola soy NOEMY MAINO

```
ES MI PRIMERA EXPERIENCIA, SALUDOS.
```

Figura 2: Comentarios en WikiMed, segunda experiencia.

A continuación se incluyen los contenidos generados en forma conjunta por los médicos participantes en cada experiencia. Estos corresponden a los artículos 'Estrategias para mejorar la comunicación entre los integrantes del equipo de salud, ante el paciente con cáncer' y 'Cómo organizar una Unidad de Tabaquismo'.



Estrategias para mejorar la comunicación *entre los integrantes del equipo de salud*, *ante el paciente con cáncer*

Calidad de la información… problemas de comunicación

Las dificultades en la comunicación entre los integrantes del equipo de salud están presentes en todos los equipos de salud, tanto de las grandes ciudades como de las pequeñas del interior. Por supuesto que el problema se agrava en las ciudades del interior o departamentos donde no existe o existe sólo un especialista, y más aún en los casos donde el paciente no es controlado por éste en el lugar de residencia sino que debe trasladarse a otra ciudad. Además de ser esto un perjuicio para el paciente, se ve afectada la comunicación entre el especialista y el médico de cabecera o médico de guardia. Esto lleva a que se plantee el primer problema de comunicación, que es la ***doble historia clínica del paciente con cáncer***.

La inexistencia de una única historia clínica que incluya los datos de control del especialista, y las dificultades para comunicarlos directamente cuando son requeridos, complican el manejo del paciente sobre todo por parte del médico de guardia o emergencia móvil, que no lo conoce y debe responder a cualquier motivo de consulta. En estos casos, puede ser un problema la ***calidad de la información que manejan tanto el paciente como la familia***, ya que nos hemos encontrado con casos de familiares que tenían una idea equivocada de la situación clínica del paciente.

En definitiva, de la disparidad de información que manejamos y del propio ***sistema de actualización de información que utilizamos en el equipo***, resultan graves problemas para la comunicación entre el equipo de salud y por ende con el paciente y sus familiares.

Nosotros ante el paciente con cáncer

Debemos decir que una actitud muy frecuente entre nosotros los médicos, es la de evitar a los pacientes con esta patología; creemos que esto se debe a la insatisfacción que nos provoca no poder "curar" al paciente. Así también, muchas veces formamos una suerte de "armadura" alrededor de nosotros mismos con el fin de no demostrar nuestras emociones y pretender no involucrarnos emocionalmente con el paciente, evitando de esta forma el sufrimiento ante el desenlace de la enfermedad.

Es interesante observar que una situación límite puede ser también una oportunidad. Cada día estamos expuestos a devenir en un tipo de atención "automatizada", y muchas veces actuamos como una computadora que recibe y devuelve información. En general, "nos olvidamos" de que tratamos con seres humanos. La presencia del fin de la existencia nos recuerda esto, y nos recuerda nuestra propia fragilidad, nuestros propios miedos, y nos invita a ser *más* humanos. Esto suele provocar que el equipo de salud tenga una actitud más sensible hacia el paciente y entre nosotros mismos, mejorando la comunicación al interior del equipo. Otras veces ocurre lo contrario, nos encerramos en una coraza para no sentir dolor y esquivamos a los pacientes en esa etapa terminal, olvidando algo muy importante: el médico también puede y debe consolar al paciente.

Rol del médico de cabecera

Reivindicamos el rol del médico de cabecera o médico de familia en el proceso de seguimiento del

paciente con cáncer. En general, es una situación gratificante y valiosa para el paciente poder contar con su médico de cabecera desde el diagnóstico hasta su agonía. Con él existe una relación privilegiada, ¿quién mejor que *su* médico comprende cuáles son sus miedos, sus aspiraciones, su plan de vida ante un futuro incierto? El paciente sabe que su médico estará acompañándolo hasta el final, y en la experiencia diaria sabemos que la palabra del médico de cabecera, su apretón de manos y su mirada alentadora en momentos difíciles será imprescindible. Los médicos constatamos que estos gestos de empatía son los que hacen que nuestro paciente colabore con las indicaciones, mejore la tolerancia al dolor y pueda reconfortar su agonía dentro de lo posible. Destacamos, complementariamente, el rol de contención del núcleo familiar que rodea al enfermo, teniendo en cuenta que este suceso vital involucra a toda la familia.

Debemos tener en cuenta lo importante que es para el paciente sentir que hay ***concordancia*** entre los planteos de su médico de cabecera, los especialistas que lo ven y los demás miembros del equipo de salud que lo asisten. Es fundamental que se sienta contenido y que no sienta que hay discrepancias entre las recomendaciones recibidas.

Muchos pacientes oncológicos no tienen médico de cabecera. Asisten a sus controles con el oncólogo y luego quedan "aislados", sin atención ni contención hasta su próximo control y, en el peor de los casos, recurren al médico de puerta quien probablemente no los conoce. ***Es fundamental que el oncólogo derive al paciente con un médico general, si es que no tiene uno; este es el primer paso para la conformación de un equipo interdisciplinario.***

La tarea de conformar un equipo

Nuestro abordaje debe ser ***integral*** cualquiera sea la patología del paciente. Como médicos debemos recordar que una enfermedad se establece en un contexto: un individuo con sus pensamientos, preocupaciones, inserto en un medio socioeconómico y cultural. Es una frase reiterada, pero aún así pocas veces implementada en la práctica clínica diaria.

El paciente oncológico –especialmente- debe recibir este tipo de atención por parte de un equipo interdisciplinario cuyos integrantes, aunque pueden variar de acuerdo al caso, suelen tener los siguientes perfiles: cirujano, oncólogo, anátomo-patólogo, imagenólogo, asistente social, psicólogo, generalista y enfermera especializada. Este equipo debería contar con instancias de discusión de pacientes; muchas veces se organizan ateneos entre cirujanos y/o entre oncólogos, pero no son tan frecuentes los ***ateneos interdisciplinarios***. La actualización de la información del paciente suele establecerse (en el mejor de los casos) a través de la historia clínica o cartas escuetas que entrega el paciente en la consulta, lo cual es totalmente insuficiente.

Nos parece importante enfatizar el ***rol del asistente social***, ya que muchas veces es el responsable de que un paciente con escasos recursos económicos pueda acceder, por ejemplo, a prótesis para la reconstrucción de órganos mutilados tal como sucede en el tratamiento de cáncer de mama, labios, conducto auditivo, entre otros, que cambian su calidad de vida y alteran el aspecto físico del paciente. Por otra parte, en Montevideo, donde las distancias también son importantes, logran obtener a través de redes sociales de apoyo, subsidios y/o bonificaciones para pacientes en tratamiento ambulatorio de quimioterapia y radioterapia, así como para repetición de medicamentos, lo que implica asistir al centro de atención varias veces en la semana. Otra carencia del sistema relacionada con lo anterior, es la ausencia del asistente social en salas de internación para informar al paciente y/o familiares de la existencia de este tipo de redes de apoyo.

En este punto del análisis, es importante resaltar la frecuente ausencia del médico de cabecera en el ***proceso de transición a la internación*** del paciente. En el mutualismo, resulta más fácil su inclusión o integración en el tratamiento dado que todos los integrantes del equipo se encuentran nucleados en un mismo sistema, lo que facilita la comunicación entre ellos. En cambio, este problema se acentúa en Salud Pública. Tomamos como ejemplo el Hospital de Clínicas, donde no hay médicos generales y las consultas que podrían ser resueltas por éstos se tratan directamente en las clínicas médicas a través de internistas. Los médicos generales se encuentran en policlínicas periféricas, siendo ésta una de las causas que dificultan su integración en el equipo tratante en este caso concreto. Lo que sucede habitualmente es que el paciente recibe su diagnóstico, inicia su tratamiento con los especialistas, y una vez finalizado vuelve a

su médico de cabecera quien recibe un ***"paciente resuelto"***, ya sea curado de su enfermedad o con un tratamiento paliativo.

Por todo esto, insistimos en la importancia de que el médico general acompañe al paciente en su proceso de enfermedad ya que es su efectivo ***conocimiento*** del diagnóstico, el estadio, pronóstico y tratamiento de la enfermedad, lo que lo habilita a resolver consultas derivadas de otros problemas, sabiendo el terreno en el que se presentan: pacientes inmunodeprimidos por quimioterapia, pacientes con somatización que deben recibir tratamiento de su afectación de base. Esto también facilita al paciente la repetición de fármacos, un asunto nada menor; una persona en Montevideo que se encuentra en tratamiento paliativo del dolor debe concurrir al Hospital de Clínicas cada pocos días, lo que implica gastos económicos, movilización del paciente o un familiar, dificultades que en gran medida podrían solucionarse si pudiera acceder a la repetición en la policlínica de su barrio.

La estrategia de *"estar contenido para acompañar"*

Son varios los elementos a tener en cuenta para generar estrategias que permitan mejorar la interrelación y la comunicación entre el equipo de salud. En primer lugar, el médico de cabecera debe tener la ***voluntad de involucrarse en el seguimiento de su paciente***, y no de hacer un pase a oncólogo y desentenderse del mismo. Las estrategias a desarrollar dependen del lugar donde cada uno trabaja, sin embargo podemos aportar ideas a la solución de los principales problemas.

El equipo de salud que atiende al paciente oncológico debe llegar a un consenso respecto a la paraclínica a solicitar. Como se dijo, muchas veces existen opiniones encontradas según el oncólogo tratante sobre los exámenes a realizar, y tanto los familiares como los pacientes solicitan del médico de cabecera su opinión e incluso la realización de exámenes específicos. Posiblemente sería de utilidad que pudiéramos generar una ***hoja de comunicación entre colegas insertada en la historia clínica***, donde el oncólogo tratante resuma para el resto de los colegas los aspectos relevantes del diagnóstico, tratamiento, pronóstico y seguimiento, especificando qué paraclínica considera importante. En esa misma ***hoja*** los colegas del equipo de otras especialidades y el médico de cabecera podrían realizar sugerencias o preguntas que serían contestadas por el oncólogo o el especialista que corresponda, en el momento de la consulta y por la misma vía.

Otro aporte que sería de mucha utilidad para la comunicación del equipo, el cual muchas veces no tiene centralizada la atención, es permitir que el paciente o el familiar del paciente lleve siempre consigo una ***copia de la historia clínica más la hoja de comunicación actualizada***, evitando lo engorroso que significa a veces conseguir una copia de la propia historia.

Otras iniciativas concretas, posibles y complementarias son las siguientes:

- Cuando se deriva el paciente a oncólogo, debe llevar una correcta historia clínica en letra legible (preferiblemente impresa) donde conste además el nombre y cómo contactar al médico de cabecera.
- El médico de cabecera debe conocer los planteos e indicaciones oncológicas registrados en la historia clínica. Si algo no queda claro se debe contactar inmediatamente al oncólogo (el teléfono móvil hace esto posible).

- Se debe incentivar la formación de equipos de cuidados paliativos, con capacitación de enfermería y médicos generalistas. Orientar estos equipos a mejorar la calidad de la asistencia al paciente y su familia, pero también al propio cuidado de cada integrante del equipo.

- Una vez que el paciente comienza el tratamiento, el médico de cabecera realizará el seguimiento, así como el equipo de cuidados paliativos y el oncólogo, todos en interrelación con una historia clínica donde se dejará todo registrado.

- Unificación de la historia clínica preferiblemente en formato electrónico. Tratar de implementar la referencia y la contrarreferencia a través de un espacio en Internet al cual puedan llegar fácilmente el médico de cabecera y los especialistas.

- El equipo de seguimiento (enfermería, nurse, médico tratante) debe tener una reunión periódica a fin de evaluar y coordinar acciones, así como plantear otras interconsultas que sean necesarias.

- Este equipo de atención primaria y oncología deberá unificar los criterios de la información que va a ser transmitida al paciente y su familia.

- Debemos contribuir a protocolizar el manejo del paciente oncológico.

Se menciona más arriba que el profesional de la salud, al enfrentar este tipo de paciente, generalmente debe elaborar conflictos internos, su propia situación ante el dolor y la muerte. Para ello es crucial el apoyo de un equipo acompañante con quien discutir situaciones, y que también pueda alertar al profesional actuante ***en ese momento en que comenzamos a irnos de nuestro rol***, siendo parte de la problemática y no de la solución. El médico de atención primaria debe tener estrecha relación con el equipo de cuidados paliativos, con la finalidad de ***estar contenido para acompañar***.

Si nos arriesgamos a ser "llaneros solitarios", seguramente en algún momento correremos el riesgo de identificarnos con situaciones, y hacer nuestras propias crisis interiores, volviéndonos ineficaces en el acompañamiento como médicos, y corriendo el riesgo de no poder resolver las crisis internas. Esto nos puede precipitar a otra situación: la de tomar distancia de estos pacientes, y no querer involucrarnos. En ambos casos es fundamental contar con colegas de la salud con quienes dialogar y quienes nos ayuden a dar los pasos necesarios para volver a un deseable equilibrio.

La conformación del equipo, con la visión bilateral de, por un lado, trabajar conjuntamente para optimizar el tratamiento, y por otro trabajar individualmente y en el ámbito grupal con la finalidad de sostenernos y enriquecernos como individuos, es fundamental para mejorar la interrelación de los profesionales actuantes. ***Necesitamos desarrollar nuestra capacidad de comunicar nuestras vivencias, y también de ser buen escucha de las vivencias del colega, sabiendo que formamos parte de un equipo que tiene una meta común, pero conscientes de que somos diferentes y en esas diferencias nos iremos enriqueciendo y desarrollando como un todo.***

CO-AUTORES DE ESTE ARTÍCULO:

Dra. Graciela Castro. Quebracho, Paysandú.

Dra. Alicia Dibello. Montevideo.

Dra. Magdalena Hermida. San José.

Dra. Rosa López. Paysandú.

Dra. Alicia Martín. Dolores, Soriano.

Br. Mariana Monteiro. Montevideo.

Dra. María Zulma Santos. Montevideo.

Dr. Aldo Silveira. Nueva Helvecia, Colonia.

Dr. Antonio Sotolani. Minas de Corrales, Rivera.

EDICIÓN Y CORRECCIÓN:

Lic. Elisa Martínez Luaces, Dra. Natalia Miranda, Ms Thais Foster (EviMed).

Ps. Soc. Amanda Sica, Dra. Isabel Alonso (CHLCC).

Abril, 2008



¿Cómo organizar una *Unidad de Tabaquismo*?

Por donde comenzar

Es importante comenzar realizando un **diagnóstico de situación de nuestra población objetivo** por área geográfica (barrio) para conocer su situación epidemiológica. Sería interesante implantar una Unidad de cesación en una zona que no cuente con la misma, o que sólo un sector de la población pueda acceder a ella (por ejemplo sólo el sector privado) a través del Centro Comunal Zonal o de un centro deportivo y/o social. Si la unidad estuviera integrada a un servicio de salud ya instalado, antes del diagnóstico de situación de la comunidad sería interesante realizar un diagnóstico de la institución.

Es imprescindible contar con apoyo institucional o de las autoridades del centro en el que va a funcionar la unidad, para asegurar la infraestructura, los recursos humanos y el respaldo necesarios para implementar las diversas intervenciones que son necesarias para un abordaje global de esta patología. También sería conveniente asegurarse desde el inicio la continuidad del servicio por al menos un año.

Como toda organización, debe dejar clara la **misión** del grupo o unidad de cesación. Sin embargo, debemos precisar que si bien su objetivo siempre será conducir al paciente a la cesación total, es posible que aún con nuestra ayuda algunos pacientes no logren ese objetivo, pero sí consolidar su motivación o avanzar en su etapa de motivación.

En cuanto al **marco teórico** para la Unidad, debe incorporar los datos epidemiológicos, la repercusión del tabaquismo en la salud pública, el concepto de tabaquismo como enfermedad adictiva y crónica. El **marco referencial** comprende: a) las políticas de salud, en el caso de nuestro país, la Ley 18.256 promulgada en marzo de 2008; b) las referencias a la institución en dónde se piensa implementar la unidad. Complementariamente, sería necesario aplicar y difundir la **Guía Nacional de Abandono del Tabaquismo** que hoy está en etapa de elaboración y aprobación.

A nivel general, nuestro objetivo mayor será realizar promoción de salud a través de acciones tendientes a promover hábitos de vida saludables y aplicar estrategias para no iniciar esta adicción dirigiéndonos especialmente a las poblaciones más fácilmente cautivas, es decir los preadolescentes, adolescentes y jóvenes. Naturalmente que el objetivo central es el estímulo a la cesación de personas fumadoras brindándoles las herramientas necesarias para cada etapa del tratamiento, así como el posterior apoyo en la etapa de seguimiento, aspecto crucial en tanto se trata de una enfermedad de fuerte tendencia a la

recaída. Sería de singular importancia ofrecer un tratamiento integral que incluya consejo nutricional, promoción de la actividad física apropiada para cada paciente, orientación para un mejor manejo de la ansiedad y uso del tiempo libre, ya que estos cambios de estilo de vida -sumados a la cesación- permiten disminuir el riesgo de enfermedades asociadas.

Al determinar los objetivos generales, los objetivos específicos y las metas en corto y mediano plazo, debemos implementar indicadores relacionados con la accesibilidad, adherencia, eficacia del tratamiento, etc. Por otra parte, la consecución de los objetivos se lograría a través de la implantación de normas estándar de tratamiento del tabaquismo. Indirectamente lograremos otros objetivos, tales como la disminución de la tasa de accidentes cerebro-vasculares, cáncer y otras enfermedades tabaco-dependientes, contribuyendo también a la salud del fumador pasivo.

Por último, como objetivo complementario para expandir la labor de la Unidad en el caso de pertenecer a un Centro de Salud, sería importante ofrecer capacitación al resto de los profesionales de la salud para que puedan actuar de forma sistemática ya que esto ha demostrado tener resultados muy favorables, especialmente con el paciente en internación.

Integración y organización de los recursos

Para organizar una Unidad de tabaquismo debemos procurar y conocer los recursos con los que se cuenta. En primer lugar los **recursos humanos**, sensibilizados con el tema, capacitados y calificados para la tarea. El equipo podría integrarse básicamente con los siguientes perfiles: médico, fisiatra, psicólogo (contención y apoyo personalizado), nutricionista, deportólogo, fisioterapeuta, personal de enfermería y educadores para la salud. También, la participación de exfumadores podría colaborar notablemente con la transmisión de la experiencia de cesación. Es claro que la ausencia de algunos de estos perfiles no debería comprometer la existencia misma o el funcionamiento de la Unidad.

El trabajo en equipo multidisciplinario procura establecer objetivos comunes y el manejo de las estrategias terapéuticas por parte de todos y cada uno de los miembros. La derivación profesional, sin embargo, es un punto importante que debe funcionar con una organización sistemática, por ejemplo a partir de un manual interno de procedimientos. El equipo debe reunirse periódicamente con el fin de conocer a cada paciente y evaluar sus logros con cada uno de ellos.

En cuanto a los **recursos materiales**, lo principal es una planta física acorde a las necesidades: consultorio equipado de forma estándar para la atención del profesional con el paciente individualmente y/o con familiar, etc.; un salón amplio para reuniones grupales; un área verde para algunas sesiones al aire libre. Es fundamental conocer los recursos económicos con los que se cuenta: si existen patrocinadores, si podemos obtener papelería gratuita a cambio de publicidad, el presupuesto para la medicación, entre otros aspectos. Sería interesante explorar el apoyo de empresas que busquen publicidad en la línea de la "responsabilidad social empresaria" para obtener recursos que permitan complementar la financiación del

Equipo multidisciplinario y los recursos del tratamiento. Por otra parte, debemos definir si nuestra Unidad responde a una organización altruista o con fines de lucro.

El consultorio requiere como **equipamiento mínimo** una camilla, un escritorio, sillas, un lavabo con jabonera, una computadora, buena iluminación, una cajonera o similar donde archivar las historias clínicas y la folletería que se le entregará a los pacientes, y finalmente un placard o vitrina con cerradura para dejar bajo llave la medicación para el tratamiento. El salón para actividades grupales debe contar con varias sillas que se aconseja ponerlas en circunferencia para la integración de todos los participantes, y luego acomodarlas en una esquina ya que ese lugar puede tener múltiples funciones, como ser usado para realizar ejercicios físicos, etc. Por supuesto, tiene que existir una antesala o sala de espera equipada según la demanda de dicha unidad y contar con servicios higiénicos.

Luego de tener todo esto resuelto, será necesario establecer con claridad que se trata de una policlínica más en un Centro de Salud, estableciendo los días y horarios disponibles para el funcionamiento e informando de ello en el lugar correspondiente a través de una cartelera que indique además la conformación del Equipo de la Unidad. Esto es fundamental para que los propios usuarios identifiquen con facilidad cuándo y cómo funciona la Unidad. Puede ser conveniente pautar las inasistencias de los pacientes, ya que la demanda puede sobrepasar las posibilidades de la Unidad y las inasistencias de los pacientes pueden dejar sin atención a otros más motivados. El encuadre del tratamiento, en la primera consulta con el paciente, estableciendo un máximo de inasistencias a la consulta que determine la suspensión del paciente en la lista de espera (dejando lugar a otro) puede ser necesario para administrar de forma efectiva los recursos de la Unidad.

En caso de que la Unidad se inserte en un Centro que ya está funcionando deberemos tener una reunión previa con todos los integrantes médicos y no médicos de dicho Centro para explicarles el modo operativo y la importancia de la Unidad, tratando de lograr una buena comunicación hacia adentro del Centro para procurar luego lo mismo hacia afuera, hacia la comunidad. El trabajo del médico y de todos los técnicos y no técnicos participantes en la Unidad de Tabaquismo debe ser retribuido como cualquier otra función sanitaria, siguiendo cierto protocolo de atención que deberá establecerse en cada institución. Es esencial la educación continua del equipo de salud abocado a esta tarea, dado que se trata de un área nueva en términos relativos, y sumamente dinámica.

Sobre la población objetivo

La población meta de una Unidad de Tabaquismo estaría conformada por 1) los pacientes fumadores ya motivados para la cesación y 2) los pacientes no motivados. Si la unidad de tabaquismo está integrada a un servicio de salud, es necesario definir y divulgar lo criterios de derivación y capacitar a los colegas para el abordaje del tabaquismo de los pacientes que no requieren de la asistencia especializada. Algunos de los subgrupos que identificamos como objetivos prioritarios serían: embarazadas, adolescentes, fumadores con patologías psiquiátricas, pacientes con patologías crónicas tabaco-dependientes (bronquitis crónica, insuficiencia respiratoria, etc.), pacientes con diabetes, pacientes con dependencia física severa o con

contraindicaciones para el uso de alguno de los fármacos específicos de primera línea. Podría abrirse un espacio complementario para familiares de estos pacientes -fumadores pasivos- en tanto el apoyo familiar es una de las causas de éxito para el fumador en su proceso de cesación, y muchas veces este apoyo no se realiza adecuadamente sino que, por el contrario, entorpece el proceso. Ofrecer información al familiar fumador pasivo para que éste pueda comprender el mecanismo de la adicción y colaborar en la tarea, sería un objetivo a priorizar.

Es de suma importancia la creación de policlínicas de cesación de tabaquismo para adolescentes y jóvenes, que son los que más se van a beneficiar, sabiendo que cuando la cesación es temprana se pueden recuperar los riesgos adquiridos casi completamente. Podemos observar cómo el consumo de tabaco cada vez cobra más fuerza en edades más tempranas; debemos ser concientes de la importancia de nuestra incidencia en esta etapa. Este grupo debe investigarse para su identificación en la Policlínica mediante algunas preguntas añadidas a la consulta médica.

La principal dificultad reside en la escasa motivación al abandono que tienen los jóvenes con un reciente inicio en el hábito; la clave es encarar este consumo como cualquier otro problema de salud que se presente en el joven, pues si no le asignamos la importancia necesaria estaremos ante un serio factor de riesgo que no hemos sabido corregir. Como subgrupo a tratar dentro de la Unidad, los adolescentes no deberían integrarse al grupo de adultos, aunque sería bueno incluir además instancias de encuentro especiales. Incorporar a la Unidad a jóvenes con pocos años de fumador es una meta que debemos imponernos para lograr, efectivamente, trabajar en la prevención de las enfermedades tabaco-dependientes.

Estrategias de trabajo

Las estrategias en este tema deben fundamentarse en la evidencia científica. El abordaje de un paciente fumador debe realizarse de una manera integral y multidisciplinaria, para alcanzar los objetivos buscados: lograr la cesación del hábito tabáquico y mantenerse en abstinencia, previniendo la recaída y si ésta sucediera su nueva cesación. También podría incluirse alguna instancia para personas que están en una actitud contemplativa frente a la posibilidad de dejar de fumar pero aún no se han decidido a comenzar un tratamiento.

La **captación** puede ser por publicidad (folletería, carteles, etc.) y por la derivación de otros profesionales que trabajen en la institución y puedan referir los pacientes que les ingresan por ese motivo de consulta. Por ejemplo, el "Espacio Adolescente" puede derivar a los pacientes que detecta con esta problemática; la policlínica de obstetricia a las embarazadas con hábitos de tabaquismo, etc. Para ello, será esencial que el Equipo de la Unidad de Tabaquismo realice la difusión de las actividades implementadas para los pacientes-objetivo de tal forma de facilitar la derivación en todo lo que sea posible.

Como parte de toda consulta es esencial que el médico indague sobre consumo de tabaco y se estimule la cesación por el consejo breve o intervención mínima, que seguramente en un buen porcentaje de casos motivará al paciente a consultar luego en nuestra Unidad.

Es importante que la Unidad esté cerca de su domicilio o trabajo dado que el tiempo que cada paciente dispone además del traslado es primordial y puede ser en muchas ocasiones parte de las resistencias que pone el paciente para dejar el tratamiento. En referencia a los pacientes del interior del país, es importante descentralizar las policlínicas para que todos tengan la misma oportunidad, ya que por lo general tienen que desplazarse a otras ciudades para tener acceso a los grupos y a la medicación.

La propuesta incluiría **actividades informativas que pueden integrar al grupo familiar del paciente**, para que ellos también puedan concientizarse de los riesgos y apoyarlo en este proceso de cambio. Sería importante informarles de que es probable que su familiar tenga recaídas, cambios en su humor (irritabilidad, ansiedad) y modificaciones en sus hábitos alimenticios, ayudarlos a fomentar actividades de distracción no asociadas a situaciones ansiógenas para el paciente, etc. El entorno en el que se mueve el paciente, la familia y su círculo social, es un factor de incidencia (positivo o negativo) en el proceso de cesación.

El tratamiento de cesación de tabaquismo en una unidad especializada incluye tratamiento farmacológico y/o cognitivo conductual. El apoyo puede ser en grupos o individual, pero siempre necesita de una asistencia personalizada. Los objetivos para el paciente deberían de prácticos y relativamente fáciles de llevar a cabo. Se debería contar con la posibilidad de realización de **ejercicio físico regular** como parte del tratamiento, ya que todos sabemos que cuando dejamos el cigarrillo, canalizamos la ansiedad hacia las ingestas, lo cual conlleva un aumento de peso que para muchos es muy preocupante. Además, el ejercicio mejorará su capacidad respiratoria y su aparato circulatorio. Como parte del equipo de trabajo, una nutricionista debería encabezar charlas educativas sobre **hábitos nutricionales saludables** y hacer hincapié en los pacientes sobre **planes de alimentación bajos en calorías**, para evitar el sobrepeso que suele acompañar a la cesación tabáquica. Este es un factor de suma importancia que condiciona el éxito, e incluso es el mayor temor de toda mujer interesada en abandonar esta adicción.

También podemos incluir el entrenamiento en **técnicas de manejo del stress y estrategias de afrontamiento** para orientar individualmente a los pacientes, con posibles derivaciones a tratamiento si esto fuera necesario. En tanto el peso de la adicción psicológica es muy fuerte, es necesario contener y orientar al paciente en un proceso de cambio que le resulta emocionalmente muy movilizante: **implica cambiar su identidad, de fumador a exfumador.** El apoyo psicológico del equipo es fundamental para orientar individualmente a los pacientes y detectar la necesidad de una derivación al equipo de salud mental si fuera necesario.

Otras estrategias pueden orientarse hacia las **actividades recreativas gratuitas** a las que los pacientes puedan acceder, solos o en grupo, como modo de habilitarlos en conductas más sanas de uso de su tiempo

libre y disociadas de los condicionamientos que tienden a relacionar con el tabaco (por ejemplo, charlas sobre distintos temas, muestras de museos, caminatas colectivas con entrega de diplomas y con la supervisión y participación del equipo de salud). Podemos también plantear a los pacientes que traigan a las reuniones grupales información sobre distintos eventos y también planificar con ellos alguna actividad mensual para así **fomentar la cohesión del grupo**, lo que redundará en un mayor apoyo ante el proyecto que ellos tienen en común (dejar de fumar y mantenerse sanos) así como un mayor grado de compromiso entre ellos, lo que potenciará a su vez nuevos vínculos y les ofrecerá una red de apoyo paralela a la del equipo profesional donde plantear su dificultades y logros en el proceso de la cesación. Es fundamental que desde el equipo se trabaje y se promueva la dinámica grupal, considerándola como otra herramienta más (y muy poderosa) para conseguir el objetivo de cesación.

El paciente debe saber que una recaída no es un fracaso y que se puede comenzar nuevamente, que los logros se irán dando paso a paso respetando los tiempos individuales de cada uno. La contención del paciente en tratamiento de cesación es esencial para evitar las recaídas; este punto es muy importante ya que el índice de recaídas es muy alto en este tema. Para la etapa de mantenimiento es necesario implementar actividades específicas, apuntando a consolidar los logros y mostrando las ventajas de la cesación. Debería incluirse el **sistema de seguimiento telefónico** (tanto en proceso de cesación como en mantenimiento) dado que ha demostrado ser un recurso muy efectivo para consolidar el trabajo del paciente, que de este modo se siente más acompañado.

Como decíamos más arriba, debemos tener en cuenta a los niños en edad escolar, quienes pueden llegar a ser los mejores promotores de salud ya que multiplican los conocimientos en su entorno. Por esto la importancia de **trabajar en forma conjunta con las maestras de la zona** donde planteamos la realización de la policlínica de cesación de tabaquismo.

Otro punto importante es el **registro de las actividades y de historias clínicas** para realizar evaluaciones periódicas de estructuras, procesos y resultados. También se deberían realizar periódicamente actividades dirigidas a los profesionales de salud que trabajan en la institución para informarles la existencia de la Unidad de Tabaquismo e insistirles en que ellos pueden realizar intervención mínima aumentando así las posibilidades de que ese paciente pueda llegar a esta Unidad.

Los obstáculos esperados y la fortaleza del equipo

Los obstáculos para llevar adelante la Unidad de Tabaquismo suelen ser muchos pero no por eso debemos frenarnos, sino seguir adelante y poner todo nuestro empeño para poder enfrentar esta problemática. Uno de los obstáculos es la dificultad que tenemos la mayoría de los trabajadores en la salud en enfrentar este problema por miedo a no saber manejarlo, lo cual nos lleva a optar muchas veces por no tomarlo en cuenta o derivarlo a un equipo más especializado.

Otras dificultades destacadas tienen que ver con la no disponibilidad de medicación gratuita, en el caso de aquellas instituciones que no tienen convenio con el Fondo Nacional de Recursos; la falta de

presupuesto para remunerar al personal de la Unidad; falta de apoyo institucional y desconocimiento de la labor del equipo en las esferas directivas; inadecuada promoción de la Unidad, incluso dentro de la propia institución. También se destaca el desánimo del Equipo ante fracasos o recaídas de los pacientes; es esencial que el Equipo esté motivado para transmitir al paciente el valor necesario para no dejar el tratamiento.

En cuanto al tratamiento en sí, luego de captado el paciente las dificultades radican en lograr la adhesión y el mantenimiento de conductas a largo plazo. Es frecuente que, ante una recaída, el paciente experimente sentimientos de vergüenza que lo inhiban de consultar al equipo de trabajo por sentir que los defraudó. Para la continuidad del tratamiento el Equipo debe lograr que el ánimo del paciente no decaiga, estimulando sus pequeños o grandes logros, sin avergonzarlo por sus fracasos. Es fundamental que su familia y amigos lo apoyen en este largo camino sin desanimarse.

En suma, nuestra tarea es encontrar la manera de que más pacientes abandonen el consumo de tabaco y superen la adicción nicotínica, haciendo del bienestar común un esfuerzo mancomunado para un logro más persistente y duradero. Debemos brindarle al fumador herramientas que lo ayuden a llenar el vacío que deja la ausencia de la caja de cigarrillos a su lado, ese elemento del que se vale para enfrentar los problemas, sentirse calmo, etc. Debemos ayudarlo a reflotar sus fortalezas internas para que ya no necesite del tabaco para enfrentar la vida.

CO-AUTORES DE ESTE ARTÍCULO:

Lic. Olga Araújo - Montevideo. Dra. Graciela Castro - Quebracho, Paysandú.

Lic. Susana Chaud - Montevideo. Dra. Alicia Dibello - Montevideo.

Dra. Eleonor Durán - Montevideo. Psic. Mariana Elgorriaga - Maldonado.

Dra. Alicia Irene Ferraro - Montevideo. Dra. Sabrina Landoni - Montevideo.

Dr. Ariel Licos - Colonia. Dra. Rosa López - Paysandú.

Dra. Noemí Maino - Tarariras, Colonia. Psic. Anna Mintz - Montevideo.

Dr. Jorge Moreale - Artigas. Dra. Lucia Nakle - Montevideo.

Dra. María Laura Paglioti - Treinta y Tres. Lic. Patricia Pinasco - Salto.

Dra. Cecilia Rivero - San Ramón, Canelones. Dra. Martha Silvera - Montevideo.

Dra. Leticia Villafán - Montevideo.

EDICIÓN Y CORRECCIÓN:

Dra. Elba Esteves Di Carlo, Lic. Elisa Martínez Luaces – EviMed.

Noviembre, 2008.

2. Bosquejo de interfaces en papel

Al iniciar el desarrollo de la aplicación *Cámara de Gesell* (*G-Sell*) se realizó un esbozo del diseño de las interfaces de usuario que compondrían el producto final. En este anexo se presentan los prototipos diseñados y validados reflejando en papel cada funcionalidad que se pretendía desarrollar y la forma en que los usuarios podrían acceder a ellas.

Se describe a continuación el prototipo creado en papel, reflejando cada funcionalidad que se pretendía desarrollar. La misma se divide en solapas cada una de ellas con funcionalidades diferentes, descriptas a continuación.

1. Solapa Emitir

Esta pestaña presenta varias funcionalidades relacionadas a la definición y emisión de entrevistas. Comienza con la interfaz presentada al usuario emisor cuando ingresa a la aplicación y desea trabajar con las entrevistas. A través de ella podrá definir una portada, comenzar la emisión o cerrar la entrevista.

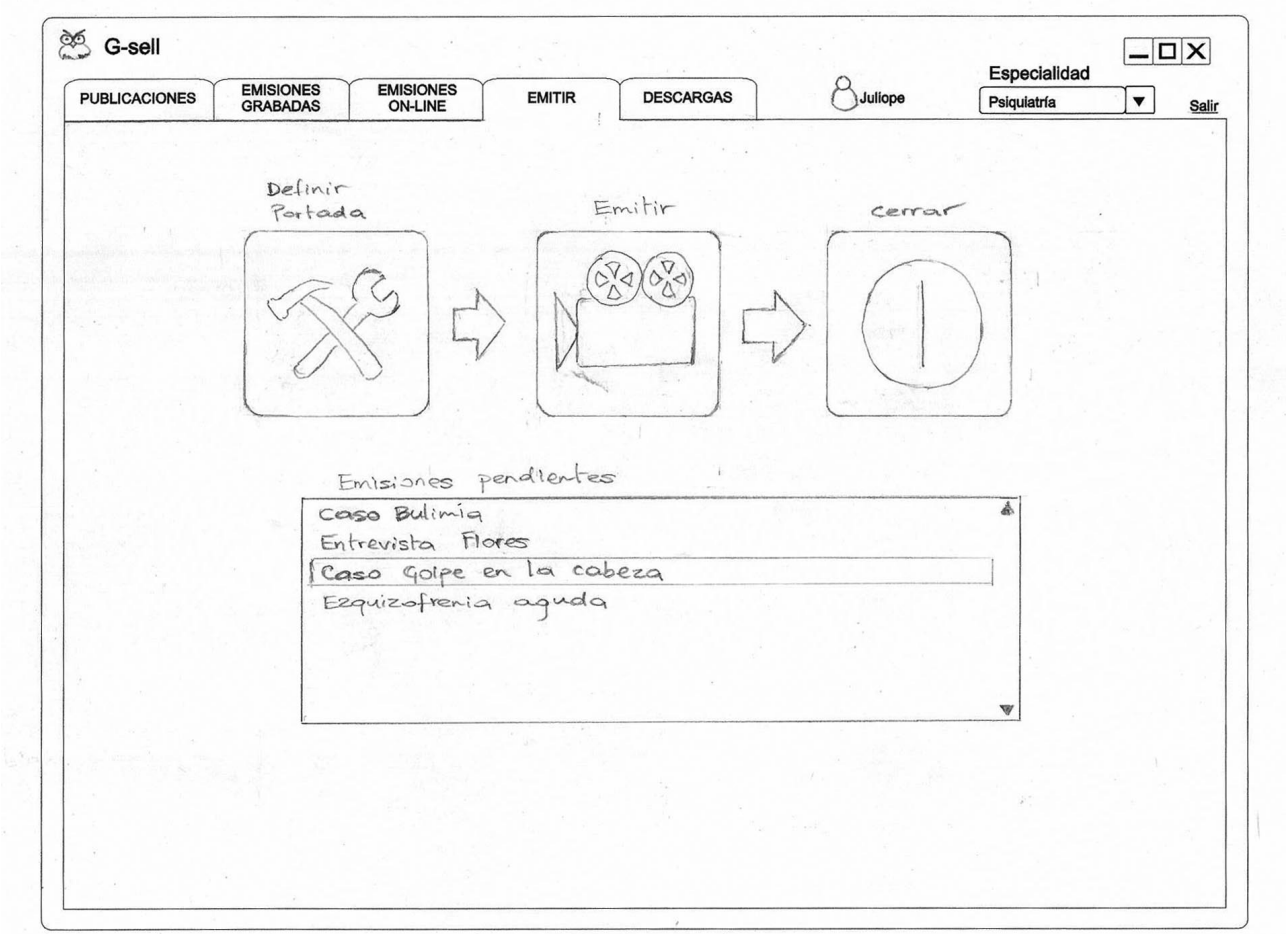


Figura 1: Inicio en solapa emitir.

Básicamente, cada grupo de funcionalidades se divide en pestañas diferentes y fácilmente identificables. Dentro de éstas, que se describen a continuación se intentó construir interfaces que fueran lo más intuitivas posible.

En el caso que el usuario elija definir una portada se le desplegará la interfaz de creación que le permitirá ingresar toda la información relevante de la entrevista, además de la lista de usuarios (colegas) a quienes podrá invitar a presenciarla.

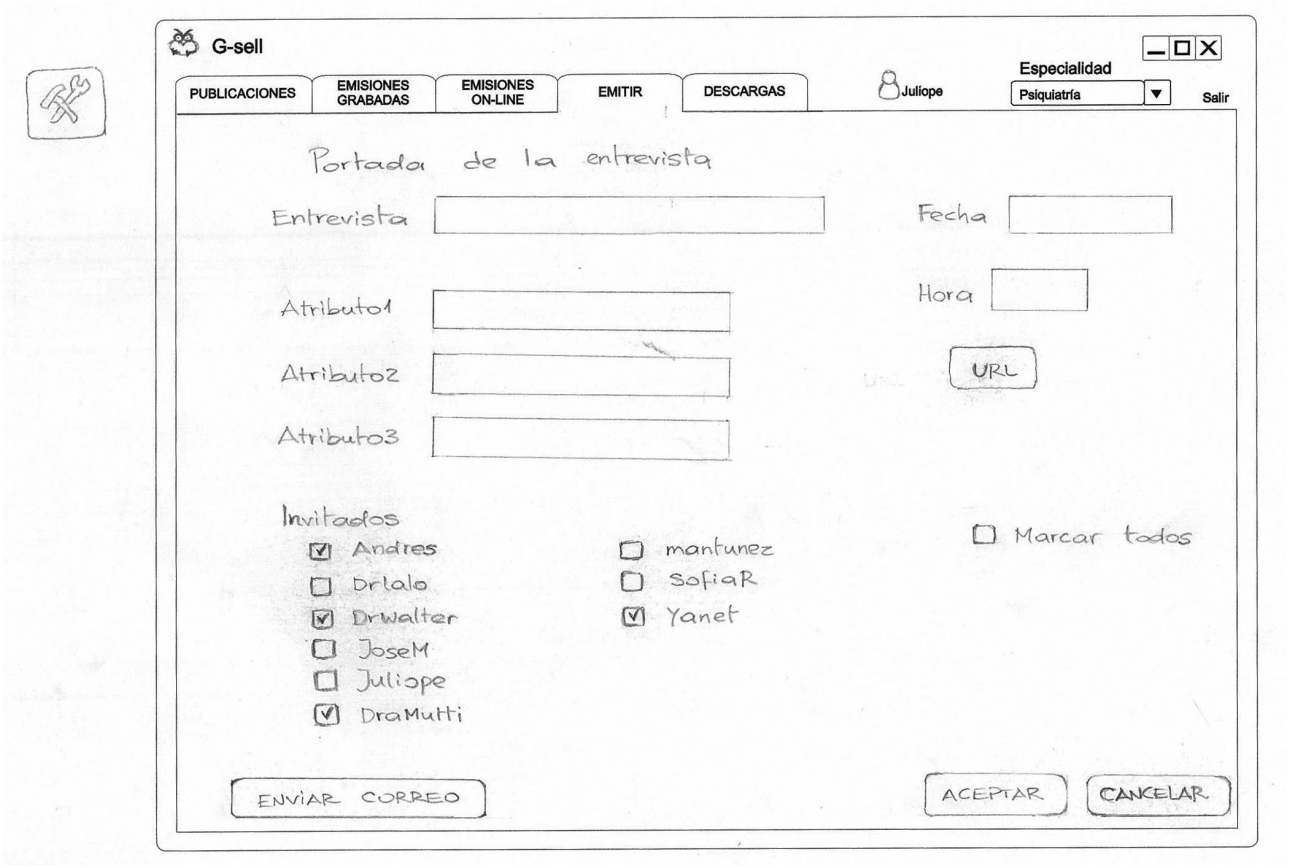


Figura 2: Creación de una portada.

Si el usuario elige el botón emitir se le desplegará la siguiente interfaz mediante la cual podría seleccionar la cámara con la cual comenzaría la emisión, así como también la ventana de chat donde podrá intercambiar comentarios con los usuarios invitados. Cabe señalar que el bosquejo de dicha interfaz fue pensada para soportar emisiones con la funcionalidad de *'Picture in Picture'*.

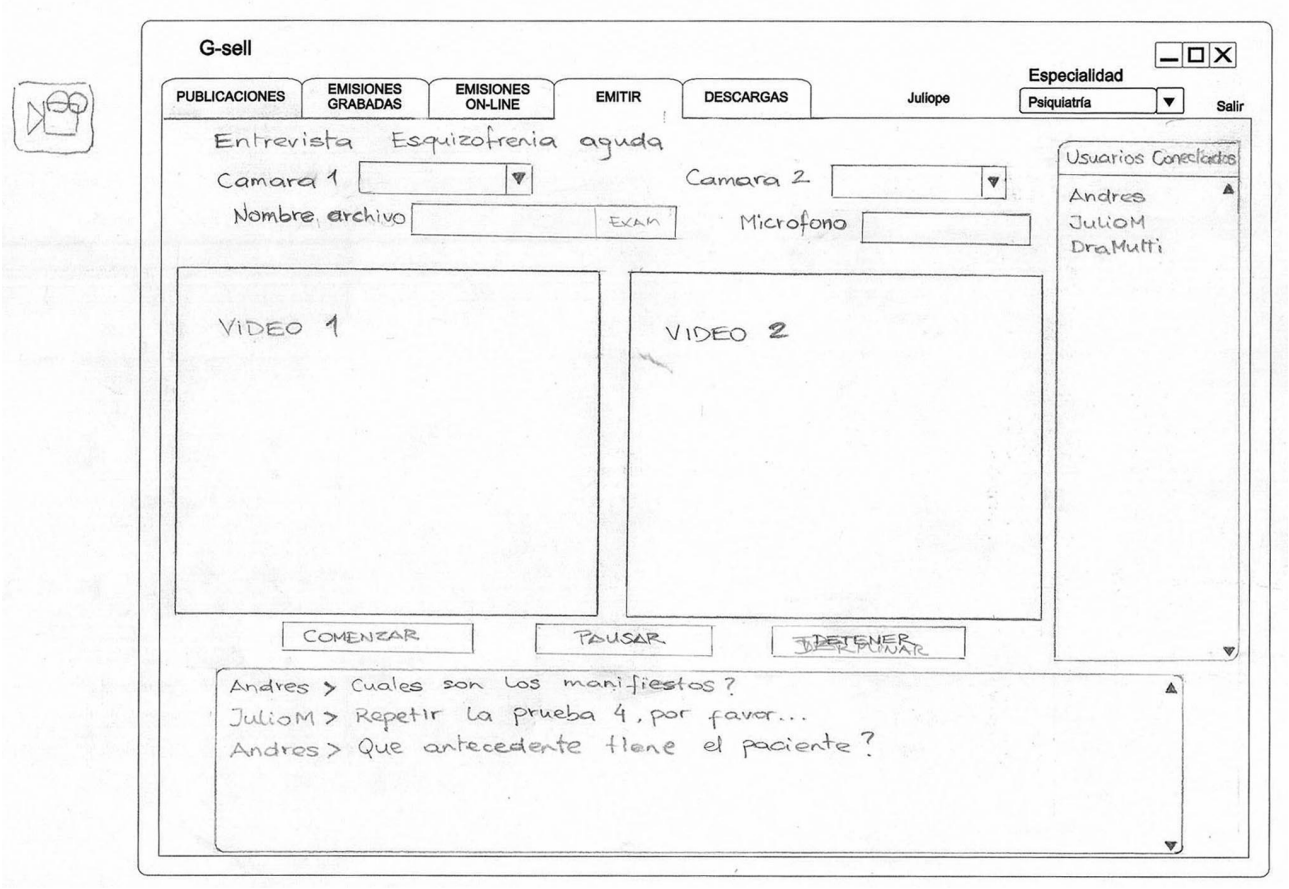


Figura 3: Comienzo de transmisión en vivo.

En el momento que el usuario elija cerrar una portada, realizará el cierre agregando información que considere pertinente y necesaria para luego publicar dicha entrevista al servidor. De esta forma, la misma pueda ser descargada y consumida a demanda por otros usuarios.

El prototipo de interfaz que se muestra a continuación es la que habilitará al usuario emisor a realizar las tareas antes mencionadas.

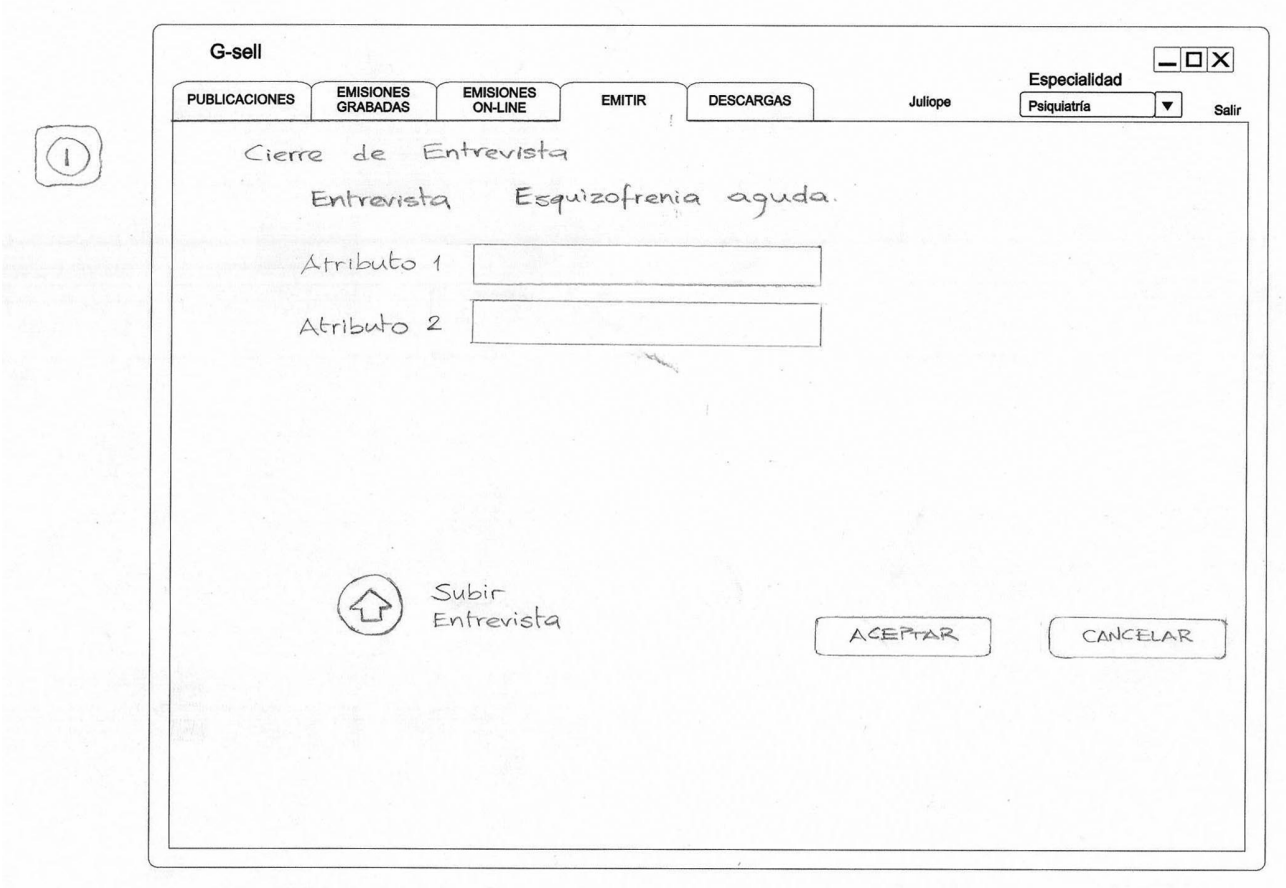


Figura 4: Cierre de entrevista.

2. Solapa Emisiones on-line

A los usuarios invitados a presenciar la entrevista en vivo se les desplegará una interfaz mediante la cual podrían observar el transcurso de la emisión llevada a cabo, así como también participar a través del servicio de chat disponible.

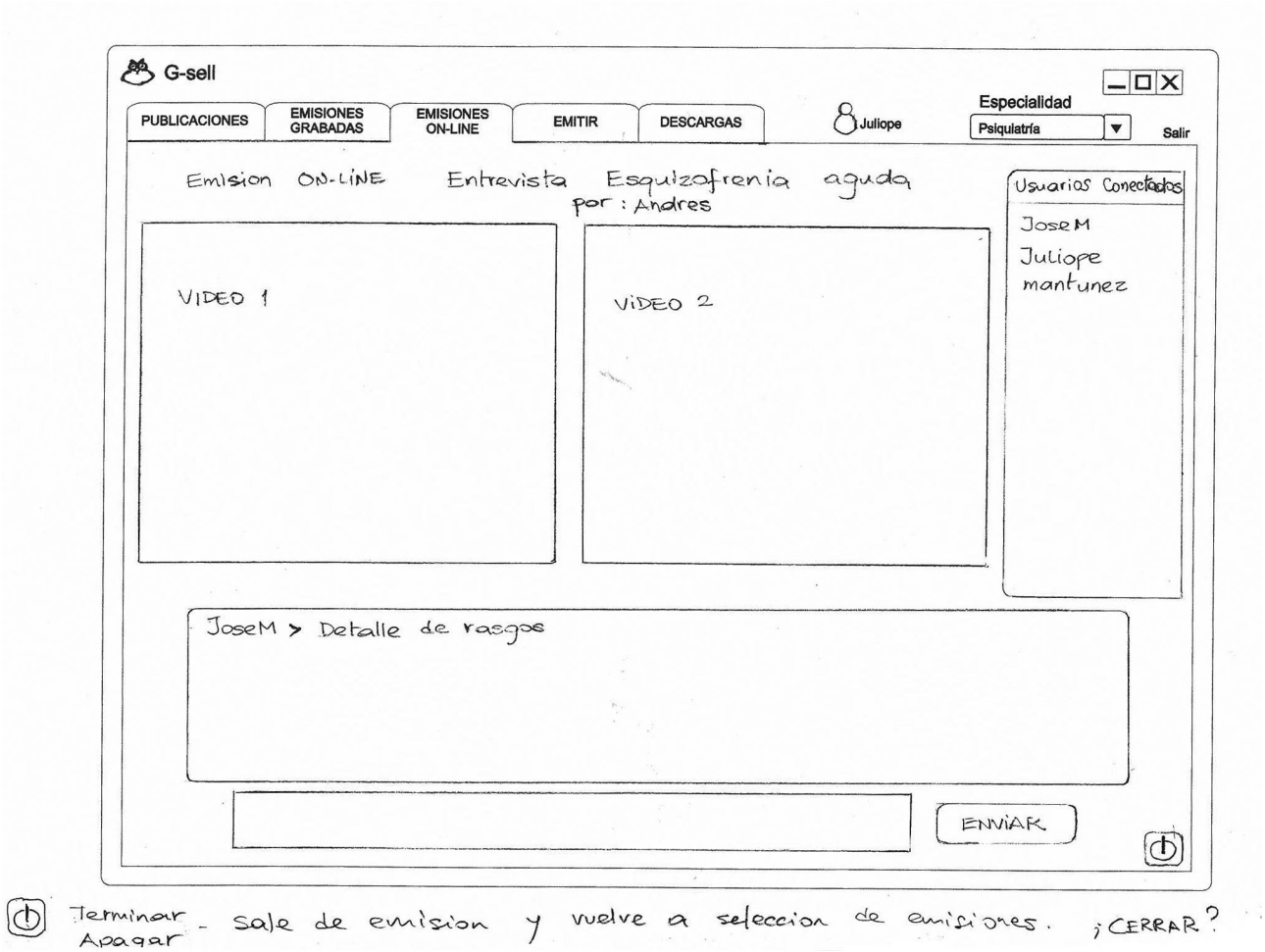


Figura 5: Consumo on-line de entrevista.

3. Solapa emisiones grabadas

Las funcionalidades proyectadas en esta solapa corresponden a cuando los usuarios invitados a una entrevista desean consumirla a demanda luego de emitida. El prototipo que modela la interfaz inicial muestra la lista de todas las entrevistas a las que el usuario fue invitado a presenciar, y por ende habilitado a descargar.

Aquí se desplegará información característica de cada entrevista, así como también una imagen previa y datos concretos que lo ayudarán de forma intuitiva y rápida a descargar la entrevista que desee.

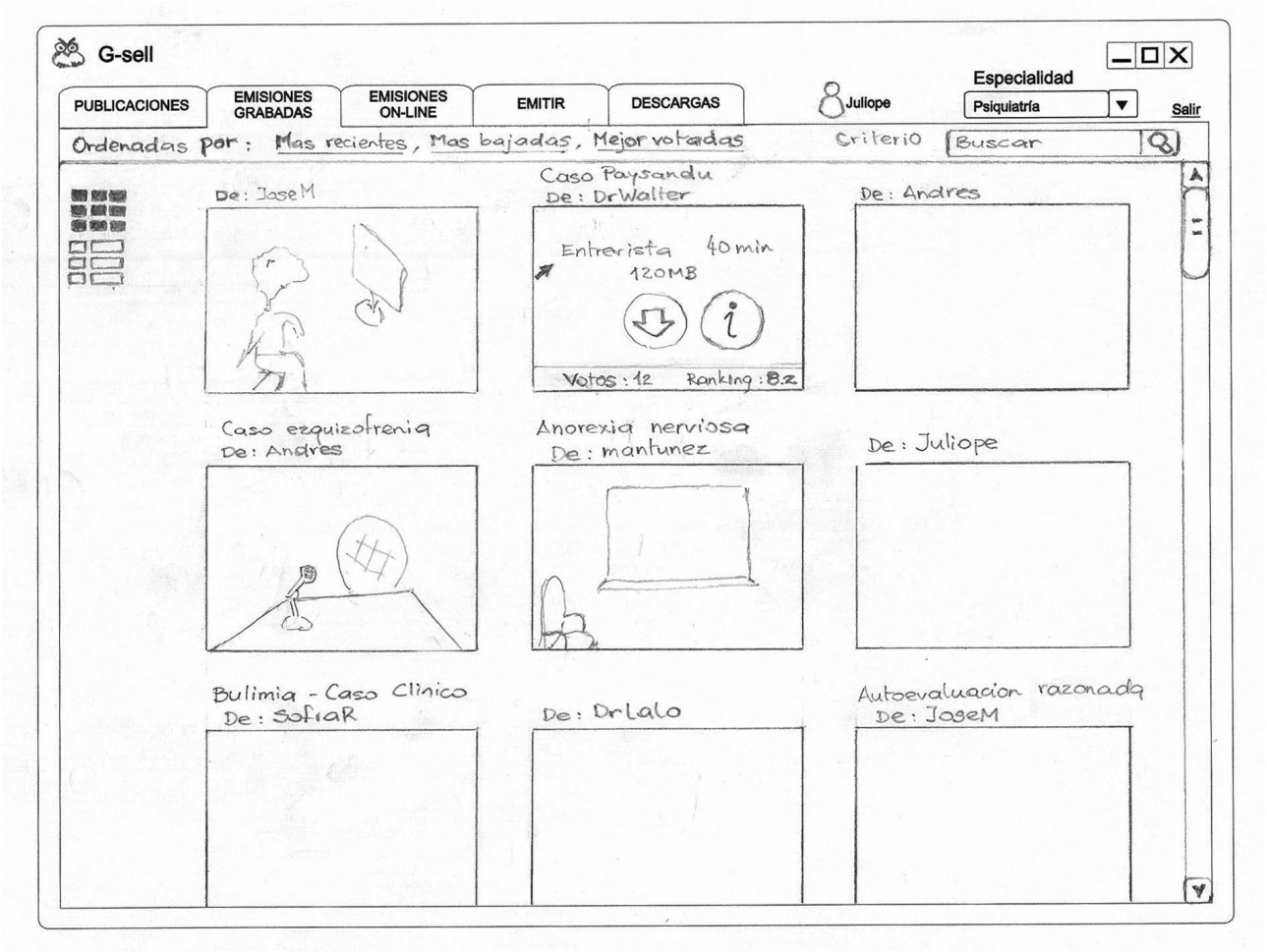


Figura 6: Emisiones habilitadas para la descarga.

El siguiente modelo en papel describe la interfaz a presentarse al usuario una vez que él haya elegido la entrevista que desea descargar. Esta pantalla contendrá la información detallada y habilita a descargarla localmente para su posterior consumo a demanda

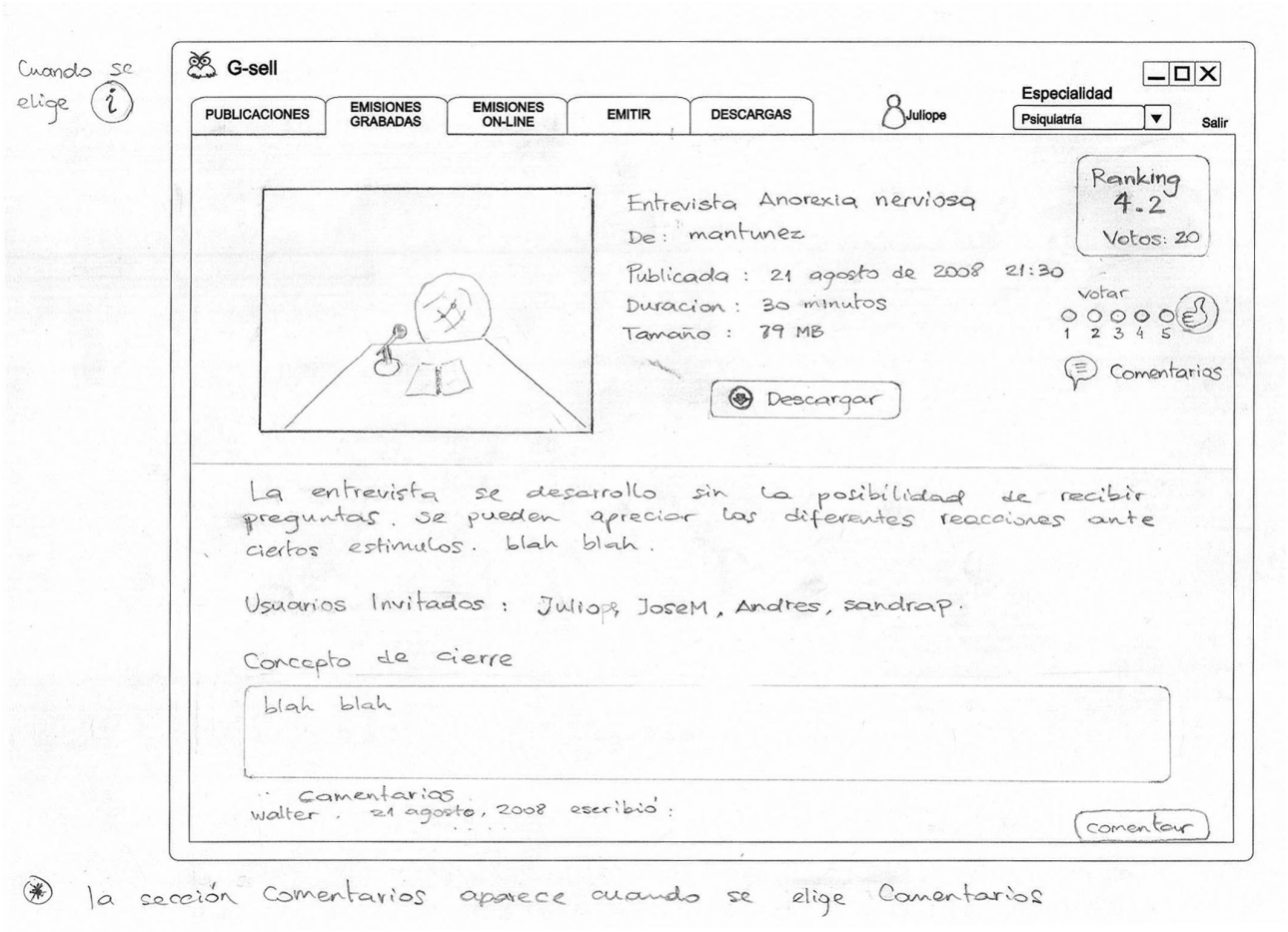


Figura 7: Selección de una entrevista determinada a descargar.

3. Internet encyclopaedias go head to head

El que sigue es un artículo de acceso pago de la revista científica Nature publicado en diciembre de 2005. Es importante su inclusión por tratarse de un estudio que aporta información valiosa en la legitimación de la inteligencia colectiva.

SPECIAL REPORT



AP PHOTO/M. PROBST

Internet encyclopaedias go head to head

Jimmy Wales' Wikipedia comes close to Britannica in terms of the accuracy of its science entries, a *Nature* investigation finds.

One of the extraordinary stories of the Internet age is that of Wikipedia, a free online encyclopaedia that anyone can edit. This radical and rapidly growing publication, which includes close to 4 million entries, is now a much-used resource. But it is also controversial: if anyone can edit entries, how do users know if Wikipedia is as accurate as established sources such as Encyclopaedia Britannica?

Several recent cases have highlighted the potential problems. One article was revealed as falsely suggesting that a former assistant to US Senator Robert Kennedy may have been involved in his assassination. And podcasting pioneer Adam Curry has been accused of editing the entry on podcasting to remove references to competitors' work. Curry says he merely thought he was making the entry more accurate.

However, an expert-led investigation carried out by *Nature* — the first to use peer review to compare Wikipedia and Britannica's coverage of science — suggests that such high-profile examples are the exception rather than the rule.

The exercise revealed numerous errors in both encyclopaedias, but among 42 entries tested, the difference in accuracy was not particularly great: the average science entry in Wikipedia contained around four inaccuracies; Britannica, about three.

Considering how Wikipedia articles are written, that result might seem surprising. A solar physicist could, for example, work on the entry on the Sun, but would have the same status as a contributor without an academic background. Disputes about content are usually resolved by discussion among users.

But Jimmy Wales, co-founder of Wikipedia and president of the encyclopaedia's parent organization, the Wikimedia Foundation of St Petersburg, Florida, says the finding shows the potential of Wikipedia. "I'm pleased," he says. "Our goal is to get to Britannica quality, or better."

Wikipedia is growing fast. The encyclopaedia has added 3.7 million articles in 200 languages since it was founded in 2001. The English version has more than 45,000 registered users, and added about 1,500 new articles every day of October 2005. Wikipedia has become the 37th most visited website, according to Alexa, a web ranking service.

But critics have raised concerns about the site's increasing influence, questioning whether multiple, unpaid editors can match paid professionals for accuracy. Writing in the online magazine *TCS* last year, former Britannica editor Robert McHenry declared one Wikipedia entry — on US founding father Alexander Hamilton — as "what might be expected of a high-school student". Opening up the editing process to all, regardless of expertise, means that reliability can never be ensured, he concluded.

Yet *Nature*'s investigation suggests that Britannica's advantage may not be great, at least when it comes to science entries. In the study, entries were chosen from the websites of Wikipedia and Encyclopaedia Britannica on a broad range of scientific disciplines and sent to a relevant expert for peer review. Each reviewer examined the entry on a single subject from the two encyclopaedias; they were not told which article came from which encyclopaedia. A total of 42 usable reviews were returned out of 50 sent out, and were then examined by *Nature*'s news team.

Only eight serious errors, such as misinterpretations of important concepts, were

©2005 Nature Publishing Group



THE *NATURE* PODCAST
Listen to Wikipedia founder Jimmy Wales talk about our survey and more at **www.nature.com/nature/podcast**

D. I. FRANKE/WIKIMEDIA FDN





Kurt Jansson (left), president of Wikimedia Deutschland, displays a list of 10,000 Wikipedia authors; Wikipedia's entry on global warming has been a source of contention for its contributors.

detected in the pairs of articles reviewed, four from each encyclopaedia. But reviewers also found many factual errors, omissions or misleading statements: 162 and 123 in Wikipedia and Britannica, respectively.

Editors at Britannica would not discuss the findings, but say their own studies of Wikipedia have uncovered numerous flaws. "We have nothing against Wikipedia," says Tom Panelas, director of corporate communications at the company's headquarters in Chicago. "But it is not the case that errors creep in on an occasional basis or that a couple of articles are poorly written. There are lots of articles in that condition. They need a good editor."

Several *Nature* reviewers agreed with Panelas' point on readability, commenting that the Wikipedia article they reviewed was poorly structured and confusing. This criticism is common among information scientists, who also point to other problems with article quality, such as undue prominence given to controversial scientific theories. But Michael Twidale, an information scientist at the University of Illinois at Urbana-Champaign, says that Wikipedia's strongest suit is the speed at which it can updated, a factor not considered by *Nature*'s reviewers.

"People will find it shocking to see how many errors there are in Britannica," Twidale adds. "Print encyclopaedias are often set up as the gold standards of information quality against which the failings of faster or cheaper resources can be compared. These findings remind us that we have an 18-carat standard, not a 24-carat one."

"Scientists' involvement would lead to a multiplier effect. Experts can help write specifics in a nuanced way."

The most error-strewn article, that on Dmitry Mendeleev, co-creator of the periodic table, illustrates this. Michael Gordin, a science historian at Princeton University who wrote a 2004 book on Mendeleev, identified 19 errors in Wikipedia and 8 in Britannica. These range from minor mistakes, such as describing Mendeleev as the 14th child in his family when he was the 13th, to more significant inaccuracies. Wikipedia, for example, incorrectly describes how Mendeleev's work relates to that of British chemist John Dalton. "Who wrote this stuff?" asked another reviewer. "Do they bother to check with experts?"

But to improve Wikipedia, Wales is not so much interested in checking articles with experts as getting them to write the articles in the first place.

As well as comparing the two encyclopaedias, *Nature* surveyed more than 1,000 *Nature* authors and found that although more than 70% had heard of Wikipedia and 17% of those consulted it on a weekly basis, less than 10% help to update it. The steady trickle of scientists who have contributed to articles describe the experience as rewarding, if occasionally frustrating (see 'Challenges of being a Wikipedian', below).

Greater involvement by scientists would lead to a "multiplier effect", says Wales. Most entries are edited by enthusiasts, and the addition of a researcher can boost article quality hugely. "Experts can help write specifics in a nuanced way," he says.

Wales also plans to introduce a 'stable' version of each entry. Once an article reaches a specific quality threshold it will be tagged as stable. Further edits will be made to a separate 'live' version that would replace the stable version when deemed to be a significant improvement. One method for determining that threshold, where users rate article quality, will be trialled early next year. ■

Jim Giles

Additional research by Declan Butler, Jenny Hogan, Michael Hopkin, Mark Peplow and Tom Simonite.
Supplementary information available online at www.nature.com/news/2005/051212/full/438900a.html

Challenges of being a Wikipedian

Vaughan Bell, a neuropsychologist at the Institute of Psychiatry in London, UK, has reworked Wikipedia's entry on schizophrenia over the past two years. Around five others regularly contribute to the reworking, most of whom have not revealed whether they have academic backgrounds. Bell says that is not a problem, as disputes are settled through the discussion page linked to the entry, often by citing academic articles. "It's about the quality of what you do, not who you are," he explains.

While admitting it can be difficult settling arguments, Bell says he often learns something by doing so. One user posted a section on schizophrenia and violence that Bell considered little more than a "rant" about the need to lock up people with the illness. "But editing it did stimulate me to look up literature on schizophrenia and violence," he says. "Even people who are a pain in the arse can stimulate new thinking."

Others, particularly those who contribute to politically sensitive entries, have found the editing process more fraught. William Connolley, a climate researcher at the British Antarctic Survey in Cambridge, has fought for two years with climate-change sceptics over the entry on global warming. When Connolley was insulted by one of the sceptics and the editing became a 'revert war' — where editors repeatedly undo each others' changes — the matter was referred to the encyclopaedia's administrators.

Two of Connolley's opponents were banned from editing any climate article for six months, but it was a bumpy process. The Wikipedia editors who oversaw the case took three months to reach a decision. They also punished Connolley for repeatedly changing the sceptics' edits, placing him on a six-month parole during which he is limited to one revert a day. Users who support Connolley have contested the decision.

"It takes a long time to deal with troublemakers," admits Jimmy Wales, the encyclopaedia's co-founder. "Connolley has done such amazing work and has had to deal with a fair amount of nonsense." **J.G.**

©2005 Nature Publishing Group

Índice de códigos y figuras

A1 – Anexos referidos al estado del Arte

1. Web Semántica....2
Figura 1: Cómo funciona la web hoy....2
Figura 2: Arquitectura de la Web semántica....4
Figura 3: Datos en la Web semántica....6
Figura 4: Mapa conceptual de la Web Semántica....7
Figura 5: Integración Web Services con Web Semántica....7
Figura 6: Cómo funciona la Web Semántica....8
2. Tendencia Web 2.0....11
Figura 1: Ejemplos de anuncios visuales....15
Figura 2: Fotos geoposicionadas....16
Figura 3: Fuentes marcadas en tag....16
Figura 4: Aplicación de la Web 2.0 en la educación....17
Figura 5: Mapa MEME....17
3. Funcionalidades Web 2.0 de Genexus....22
Figura 1: Entorno de desarrollo gráfico....22
Figura 2: Control para gráficas....23
Figura 3: Control para información jerárquica....24
Figura 4: Control para imágenes y texto....24
Figura 5: Crear control, definición....25
Figura 6: Crear control, propiedades....26
Figura 7: Crear control, diseño....27
Figura 8: Crear control, ejecución....28
4. Estudio de iTunes U....30
Figura 1: Organización jerárquica del sitio....32
Figura 2: Eventos disparados al requerir contenido....34
Figura 3: Portada de Universidad de Berkeley en iTunes U....36
Figura 4: Listado de contenido disponible....37
Figura 5: Vista particular de la aplicación iTunes....39

A2 – Anexos referidos a tecnologías y patrones de diseño

2. Modelo-Vista-Controlador....44
Figura 2: Flujo de control....45
3. Introducción a Windows Presentation Foundation (WPF)....46
Figura 1: Tabla plataforma unificada para interfaces de usuarios modernas....47
Figura 2: Inclusión de publicidad en la aplicación....47
Código 1: Ejemplo XAML de botones en pestaña Emisiones por Demanda....48
Figura 3: Interfaz de usuario....48
Código 2: Ejemplo XAML, evento del botón Play_Click....48
Código 3: Implementación del ejemplo anterior....49
Figura 4: Resultado del ejemplo....49
Código 4: Ejemplo XAML de control MediaElement....51
Figura 5: Ejecución de control MediaElement....51
Código 5: Ejemplo XAML de control StackPanel....52
Figura 6: Ejecución del control StackPanel....52
Código 6: Ejemplo XAML de control DockPanel....53
Figura 7: Ejecución del control DockPanel....53
Código 7: Ejemplo XAML de control Canvas....54
Figura 8: Ejecución del control Canvas....55
Código 8: Ejemplo XAML de control Grid....55
Figura 9: Ejecución del control Canvas....56
Código 9: Ejemplo XAML de control FlowDocumentPageViewer....57
Figura 10: Ejecución del control FlowDocumentPageViewer....57
Código 10: Ejemplo XAML del control Tab....58

Figura 11: Ejecución del control Tab....58
Código 11: Ejemplo XAML del control Wrap panel....58
Figura 12: Ejecución del control Wrap panel....59
Código 12: Binding de control con datos....59
Código 13: Ejemplo XAML del control de estilos....60
Figura 13: Ejecución de estilos....60
Código 14: Ejemplo XAML de asignación de estilos....61
4. Introducción a Windows Communication Foundation (WCF)....62
Figura 1: Conceptos WCF....63
5. Publicación y descarga de archivos....65
Figura 1: Referencia FileTransferClient....65
Código 1: Configuración de FileTransferService en app.config....66
Figura 2: Diagrama de clase StreamWithProgress....67
Código 2: Reimplementación de método Read en StreamWithProgress....67
Código 3: Sección de método UpLoadZipFile....68
Código 4: Sección de método EmpaquetarEmision....68
Código 5: Sección del evento DownloadButton_Click....69
Código 6: Sección del método DownLoadFile....69
Código 7: Sección del método OpenFile....69
Figura 3: Sección de configuración en Web.config....70
Código 8: Vista de implementación del contrato....71
Figura 4: Definición de FileTransferService....71
Código 9: Implementación de DownloadFile....72
Código 10: Implementación de Dispose en RemoteFileInfo....72
Código 11: Sección de método UpLoadFile....72
Código 12: Implementación de la extracción de JPG....73
Figura 5: Estructura de servicio Web FileTransfer....73
6. ChatMessage....77
Código 1: Clases y declaración de array para creación de archivo XML....77
Código 2: Estructura de cola utilizada....78
Código 3: Implementación método AddMessage....78
Código 4: Implementación método Save....78
7. Servicios dúplex....79
Código 1: Declaración ServiceContract con *callback*....80
Código 2: Implementación de Interfaces....80
Código 3: Archivo *svc* WCF....80
Código 4: Archivo de configuración del servicio WCF....81
Figura 1: Generación proxy de un servicio *svc*....82
Código 5: Uso de proxy WCF en aplicación cliente....82
Código 6: Implementación *callback* de un servicio WCF....82
Código 7: Archivo de configuración de aplicación cliente WCF....83
8. Chat del canal del mismo nivel....84

A3 – Anexos referidos a Base de Datos

2. Acceso a datos....93
Figura 1: Arquitectura de acceso a datos....93
Figura 3: Clases de Linq to Sql....96
Figura 4: Diseñador relacional de objetos....97
Código 1: Archivo de configuración....98
Figura 5: Archivos del componente Gesell.DataAccessLayer....99
Código 2: Clase parcial....99
Figura 6: Métodos de clase entidad....100
Figura 7: Vista de clases del componente Gesell.DataAccessLayer....101
Figura 8: Subcapas de acceso a datos....101
Código 3: Obtener datos, caso simple....102
Código 4: Obtener datos, caso complejo....103
Código 5: Alta de datos....104
Código 6: Alta de datos transaccional....104

Código 7: Modificación de datos, marcas de tiempo.....105
Código 8: Modificación de datos, valores originales de propiedades de entidad.....105
Código 9: Modificación de datos, entidades completas original y modificada.....106
Código 10: Eliminar datos.....106
Código 11: Seleccionar entidad.....107
Código 12: Seleccionar entidad usando join.....107
Código 13: Ejecución de sentencias sql.....107
Código 14: Funciones.....107
3. Componente Local.....109
Figura 1: Base de datos local en explorador de servidores.....110
Código 1: Insertar registro.....110
Código 2: Eliminar registro.....111
Código 3: Obtener registro.....111
Figura 2: Capa de acceso a datos.....114
Figura 3: Diagrama de clases.....115

A4 – Anexos referidos a video y streaming

1. Captura y streaming multimedia.....118
Código 1: Inizialización del grafo.....118
Código 2: Agrega filtros de captura y entrega.....119
Código 3: Agrega filtros de copia de flujo.....119
Código 4: Conexión manual de pines de entrada y salida.....120
Código 5: Conexión automática de filtros.....120
Figura 1: Grafo DirectShow de captura.....121
Figura 2: Sección del archivo WMSysPr9.prx.....122
Código 6: Configuración de filtros en formato ASF.....123
Código 7: Conexión de filtro para grabación.....123
Código 8: Definición y conexión de filtro para emisión.....124
Código 9: Definición de objeto PushSink para emisión.....124
Código 10: Implementación de AcquireCredentials en clase Autentica.....125
Código 11: Preparación de comunicación con WMS.....125
Figura 3: Grafo DirectShow de emisión.....126
Figura 4: Vista de clases de MediaKit modificada.....127
Figura 5: Vista parcial del diagrama de clases.....127
Figura 6: Vista completa de clase CaptureElement.....128
Código 12: Implementación de DecideChangeGraph.....128
Código 13: Implementación del PlayToStream.....128
Figura 7: Vista completa de clase VideoCapturePlayer.....129
Código 14: Modificación en la implementación de método Play.....129
Código 15: Implementación de StopStreamToServer.....130
Figura 8: Directiva de definición para MediaKit en WPF.....130
Código 16: Definición de control CaptureElement en XAML.....130
Código 17: Declaración de control en C sharp.....131
Código 18: Configuración del formato de la muestra e inicio de captura.....131
Código 19: Obtención de amplitud promedio en instante x.....132
Código 20: Fórmula para pasaje de muestras promedio a dB.....132
Código 21: Creación de imagen JPG.....132
2. Multimedia en Windows.....134
Figura 1: Estructuras de comunicación entre aplicación y hardware.....134
Figura 2: Capas HEL y HAL.....136
Figura 3: Grafo para captura de cámara web.....139
Figura 4: Estructura de AM_MEDIA_TYPE.....140
Figura 5: Estructura Info de video.....140
Figura 6: Estructura de VIDEOINFOHEADER.....141
Figura 7: Protocolo de conexión de filtros.....142
Figura 8: Protocolo de streaming.....143
Figura 9: Vista parcial del diagrama de clases de MediaKit.....148
Código 1: Objeto DvdPlayerElement en XAML.....148

Código 2: Objeto VideoCaptureElement en XAML.....149
3. Formato de archivos multimedia.....150
Figura 1: Windows Media 9 vs. Windows Media 8.....152
4. Servidor de streaming.....156
Figura 1: Comparación de protocolos TCP y UDP.....159
Figura 3: Firewalls bloquean todo tipo de tráfico, excepto el HTTP.....162
Figura 4: Se encapsula streaming en HTTP.....162
Figura 5: Emisión de flujo por unicast.....164
Figura 6: Esquema de distribución por multicast/unicast.....164
Figura 7: Asistente de creación.....166
Figura 8: Configuración de protocolos de entrega.....167
Figura 9: Configuración de autorización ACL.....168
Figura 10: Asignación de permisos al usuario genérico.....168
Figura 11: Buffer de tiempo.....169
Figura 12: Composición de retardo por buffers.....170
Figura 13: Desactivar el buffer de WMS.....171

A5 – Anexos referidos a los resultados

1. Experiencia en WikiMed.....174
Figura 1: Comentarios en WikiMed, primera experiencia.....175
Figura 2: Comentarios en WikiMed, segunda experiencia.....175
2. Bosquejo de interfaces en papel.....187
Figura 1: Inicio en solapa emitir.....187
Figura 2: Creación de una portada.....188
Figura 3: Comienzo de transmisión en vivo.....189
Figura 4: Cierre de entrevista.....190
Figura 5: Consumo on-line de entrevista.....191
Figura 6: Emisiones habilitadas para la descarga.....192
Figura 7: Selección de una entrevista determinada a descargar.....193

Universidad de la República

Facultad de Ingeniería
Instituto de Computación
Montevideo – Uruguay

Proyecto de Grado

Aplicación de redes sociales a la educación a través de Internet.

MANUAL DE USUARIO

SISTEMA G-SELL



Alejandra Borghi
Yanet Cortés
Walter López Da Cunha

Tutores:
Martín Furno
Pablo Rebufello

Informe de Proyecto de Grado presentado al Tribunal Evaluador como requisito de graduación de la carrera Ingeniería en Computación.

Índice de contenidos

1. Inicio....2
2. Publicaciones....2
    2.1 Alta de Enlace....3
    2.2 Alta de Publicación....4
3. Emisiones Grabadas....5
4. Emisiones On-Line....6
5. Emitir....7
    5.1 Nueva Entrevista....8
    5.2 Modificación de Portada de Entrevista....9
    5.3 Agendar....9
    5.4 Emitir....10
    5.5 Cerrar....11
    5.6 Publicar....12
6. Descargas....13
7. Administración....15
Índice de ilustraciones....19

Manual de usuario

Se describen en este documento las funcionalidades más relevantes del sistema G-Sell y como el usuario puede acceder a ellas.

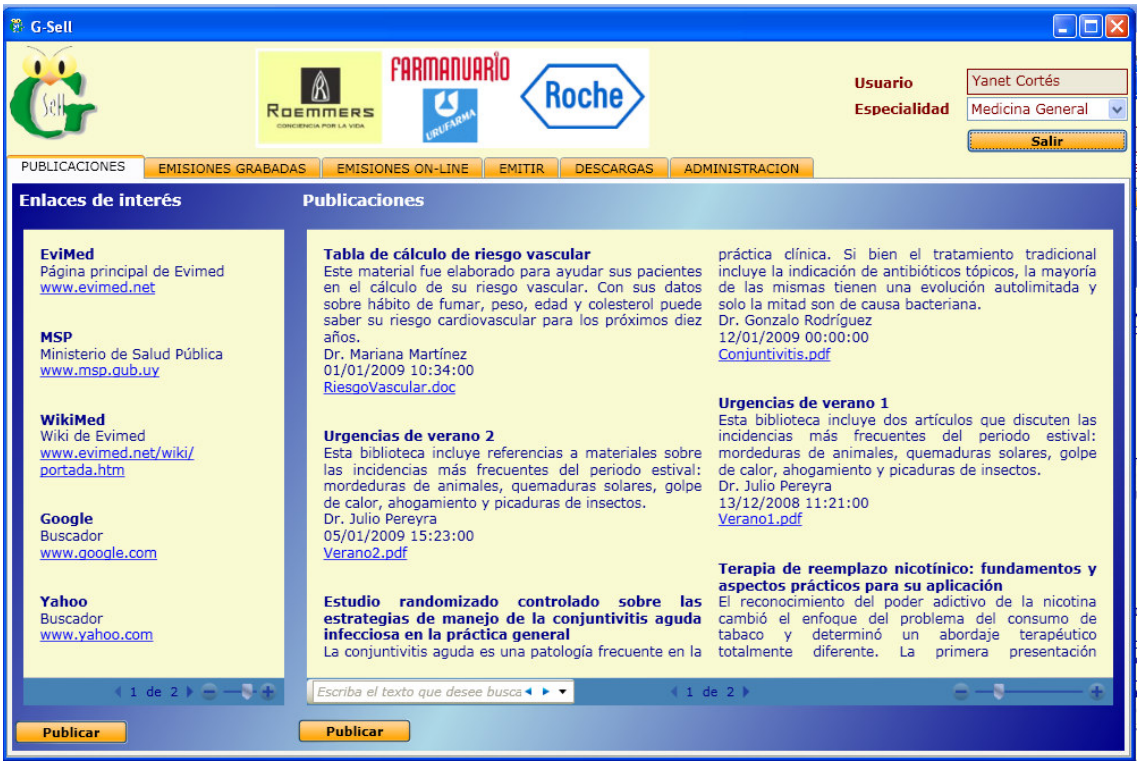


Figura 1: Pestaña Publicaciones.

La interfaz de usuario consiste en una ventana que contiene un control que ocupa su mayor parte y que proporciona el acceso a las pestañas que conforman el acceso principal a todas las funcionalidades del sistema: *Publicaciones, Emisiones Grabadas, Emisiones On-Line Emitir, Descargas* y *Administración*. Luego en cada pestaña se definen diferentes paneles que permiten navegar en las funcionalidades resueltas en cada pestaña.



Figura 2: Espacio reservado.

En la parte superior de la ventana se encuentra un espacio reservado. En su parte izquierda se visualiza el logo del sistema G-Sell, en el cual se puede entrever la lechuza asociada al programa. Dado que el sistema permite, entre otras cosas, visualizar transmisiones de video, la lechuza resultó un fiel representante. En la parte superior derecha se cuenta con controles que permiten al usuario registrarse en el sistema. Además de ingresar datos de usuario y contraseña, debe seleccionarse una de las especialidades que tenga asociada. El resto de la zona superior es utilizada para propaganda. Los patrocinantes o colaboradores del sistema podrían proporcionar información respecto a ellos para ser publicada en este lugar.

1. Inicio

Inicialmente, cuando comienza la ejecución del programa, si se establece la conexión con el servidor de aplicaciones, se carga la pestaña *Publicaciones*. En caso de que no sea posible la conexión, se estaría frente al uso del sistema en modalidad desconectado. Las funcionalidades disponibles serían las locales del Smart Client. Solamente la pestaña de *Descargas* permite el acceso a estas funcionalidades.

La única pestaña que puede proveer información sin que se registre un usuario es la de *Publicaciones*. Se permite visualizar las *Publicaciones* y *Enlaces de interés* que fueron publicados por los usuarios del sistema.

El resto de las pestañas permanecen deshabilitadas hasta que el usuario se registra. La mayoría de las funcionalidades del programa dependen del usuario registrado y otras de la especialidad seleccionada. Cuando el usuario se registra, se carga por defecto una especialidad. Si el usuario tuviera más de una asociada, podrá modificar su elección.

2. Publicaciones

El panel principal de la pestaña de *Publicaciones* está divida en dos secciones. La de la derecha permite visualizar las publicaciones realizadas por todos los usuarios. Éstas tienen un título asociado, una descripción breve de la publicación e información de los autores, además fecha y hora en que se publicó la misma en el sistema. También se cuenta con un enlace a un documento que contiene la publicación propiamente dicha. Este documento puede ser consumido por cualquier usuario del sistema. Al seleccionar el enlace se podrá abrir el documento o descargarlo localmente.

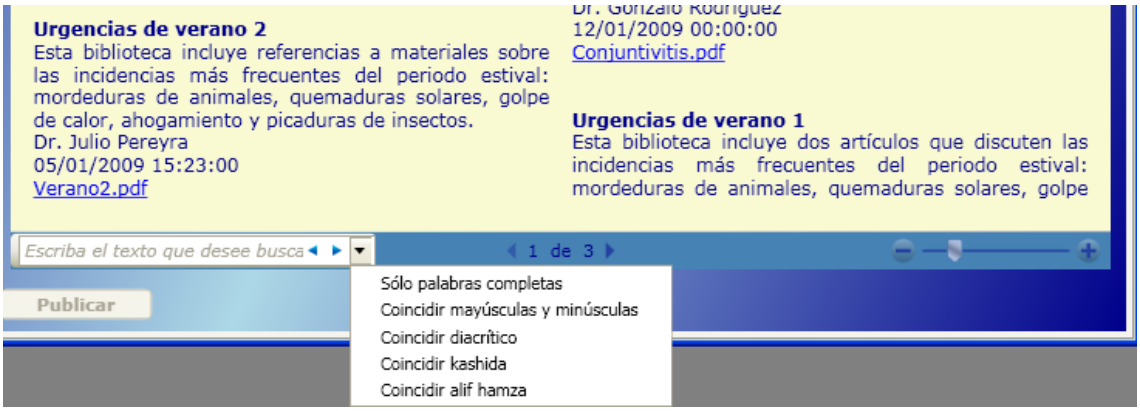


Figura 3: Publicaciones.

En la parte inferior del control que provee las publicaciones se cuenta con otro control que permite navegar entre las páginas necesarias para contener toda la información de las publicaciones existentes. Se tiene visible el número de página y se pueden utilizar las flechas derecha e izquierda para navegar entre las mismas.

En la misma parte inferior, más a la derecha se tiene un control que permite ampliar o achicar la vista de las páginas y modificar el tamaño de la tipografía utilizada. La información que no pueda ser incluida en la página cuando se aumenta el tamaño pasa a la página siguiente, y por el contrario, cuando se reduce. Cuando se disminuye el tamaño del texto se puede visualizar información que inicialmente estaba en otra página.

En esa zona pero del lado izquierdo se tiene un control que permite buscar texto en los resúmenes de las publicaciones. Luego de ingresar el texto se podrá seleccionar las flechas que permiten buscar la siguiente o la anterior instancia del texto que se esta buscando. Se cuenta con algunas opciones más específicas de búsqueda al seleccionar el botón *Buscar* del mismo control.

La parte izquierda de la pestaña es similar a la zona de *Publicaciones*. Allí se visualizan los *Enlaces de interés*. Estos también son publicados por los usuarios. Consisten en un identificador del enlace, una descripción y el enlace que permite acceder a la página o sitio que se referencia. Al igual que en las publicaciones, también cuenta con controles en la parte inferior que permiten cambiar el tamaño del texto o navegar entre los enlaces.

Cuando el usuario selecciona alguno de los enlaces, ya sea los que referencian a los archivos que tienen la publicación o a sitios de interés, se despliega el *Navegador* de *G-Sell*.



Figura 4: Navegador.

El usuario puede navegar en las opciones de la página que se visualiza o puede indicar una nueva dirección a abrir.

En la parte inferior del panel principal de la pestaña *Publicaciones* ambas secciones tienen un botón *Publicar* que permite acceder al panel correspondiente para realizar el alta de las *Publicaciones* ó de *Enlaces de interés*.

2.1 Alta de Enlace

Para poder publicar un *Enlace de interés* deben especificarse título, comentario y enlace al sitio. Luego de aceptar o cancelar los cambios se vuelve al panel principal de la pestaña *Publicaciones*.



Figura 5: Alta y modificación de Enlaces.

2.2 Alta de Publicación



Figura 6: Alta y modificación de Publicaciones.

Al ingresar una nueva publicación debe indicarse título, comentario a modo de resumen de la publicación y los autores de la misma. En este caso se debe asociar el documento que se encuentra en el equipo local, que quiere publicarse. Se puede seleccionar el archivo a través

del botón correspondiente. Cuando se confirme el alta de la publicación este archivo será subido al servidor de archivos y quedará disponible para ser consumido por otros usuarios en el panel principal de la pestaña de *Publicaciones,* como se indicó anteriormente. Se vuelve a dicho panel luego de aceptar o cancelar los cambios.

3. Emisiones Grabadas

La pestaña *Emisiones Grabadas* muestra las entrevistas publicadas disponibles para ser consultadas o descargadas por el usuario. Estas entrevistas se filtran por la especialidad seleccionada por el usuario en la interfaz. Solamente se incluyen las entrevistas a las que el usuario ha sido invitado a presenciar.



Figura 7: Emisiones Grabadas.

De cada entrevista se muestra su descripción, el usuario responsable, la cantidad de descargas realizadas, el ranking de votaciones y una imagen previa del video. Haciendo doble click sobre una de las entrevistas se accede a la entrevista en forma individual.

Allí se visualiza toda la información asociada a la entrevista: imagen previa del video, descripción, usuario responsable y fecha en que se llevó a cabo la emisión, entre otros. También se visualizan los datos de la portada y de cierre de la entrevista.

A través del botón *Descargar* se permite descargar el video y la información asociada a la entrevista al disco local del usuario. Luego podrá ser accedida localmente a partir de la pestaña *Descargas* y consumida sin necesidad de conexión.

Con el botón *Cerrar* se vuelve al panel principal de la pestaña *Emisiones Grabadas*.



Figura 8: Emisión Grabada.

4. Emisiones On-Line



Figura 9: Emisiones On-Line.

En la pestaña de *Emisiones On-Line* se encuentra una lista con las emisiones agendadas del usuario. Esta lista se genera a partir de las invitaciones realizadas por otros usuarios para asistir a la emisión en vivo de entrevistas. Para poder conectarse a la emisión en vivo se debe seleccionar la emisión y utilizar el botón *Ver Emisión*.

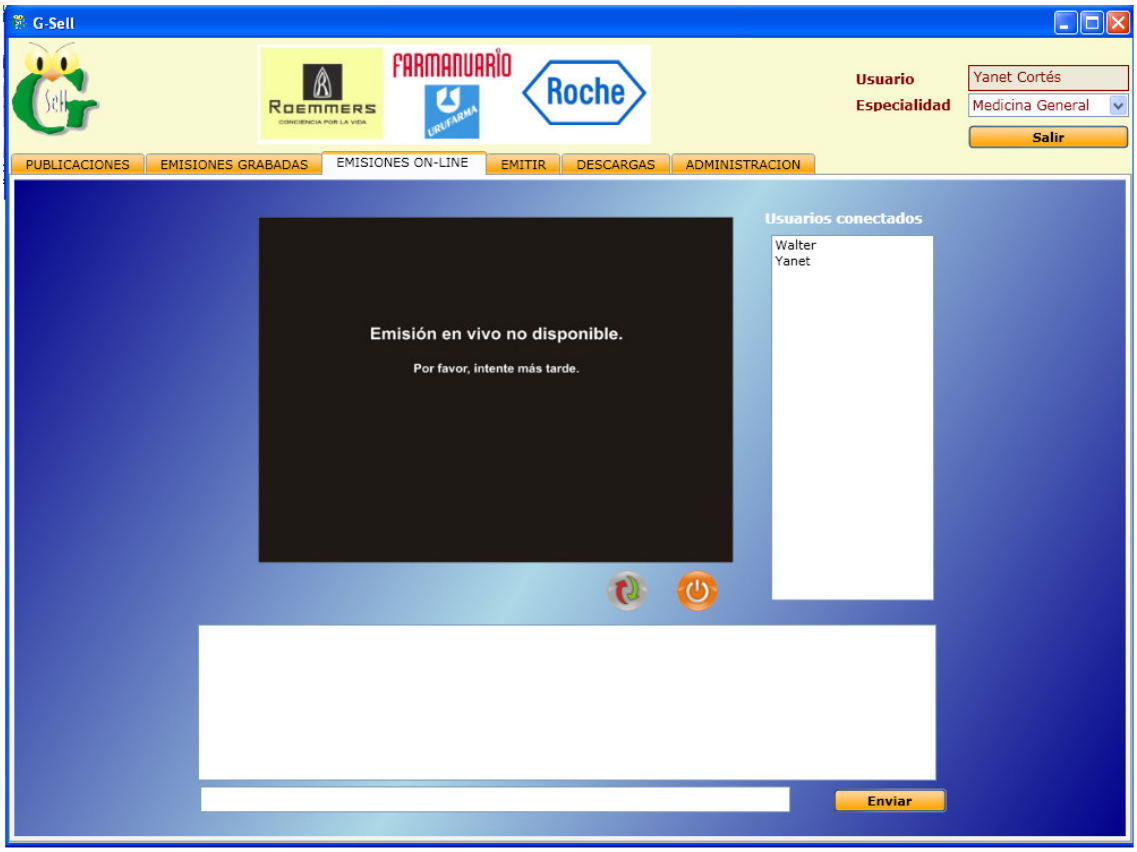


Figura 10: Emisión en vivo.

Se accede al panel que permite ver la emisión en vivo que otro usuario esta compartiendo. Si la emisión del video no ha comenzado aún se visualiza una imagen pidiendo al usuario que intente conectarse más tarde. Del lado derecho se podrá ver quienes están ya conectados a la emisión. En los controles de abajo se podrán ver los mensajes intercambiados durante la emisión y podrá enviar sus propios mensajes usando el botón *Enviar*.



Figura 11: Botones de Emisión en Vivo.

El botón de la izquierda, *Recargar Emisión,* permite reintentar la conexión a la emisión del video. El botón *Terminar*, finaliza la conexión y la interfaz vuelve al panel principal de la pestaña *Emisiones On-Line*.

5. Emitir

En la pestaña *Emitir* se tienen los puntos de entrada para las diferentes funcionalidades asociadas al ciclo de vida de las entrevistas. Se visualizan botones a partir de los cuales se disparan los diferentes procesos y una lista de entrevistas pendientes. Las entrevistas que se ven son las asociadas al usuario registrado en el sistema. La especialidad de las entrevistas puede ser cualquiera de las especialidades asociadas al usuario.



Figura 12: Pestaña Emitir.

Las entrevistas tienen un estado que indica en qué parte de su ciclo se encuentran. Los estados que se utilizan, ordenados según los procesos que se van cumpliendo son: *Definir Portada, Agendar, Emitir, Cerrar, Publicar* y *Finalizada*.

Según el estado de la entrevista seleccionada, se habilitan los botones que disparan las funcionalidades que se pueden aplicar. Los botones que se incluyen, de izquierda a derecha permiten:

- Iniciar una nueva entrevista ó Definir portada de la entrevista.
- Agendar la entrevista.
- Emitir la entrevista.
- Cerrar la entrevista.
- Publicar entrevista.

5.1 Nueva Entrevista

El  primer botón sirve tanto para iniciar una entrevista nueva como para definir la portada de una entrevista ya existente. Para usar la funcionalidad de nueva entrevista, debe seleccionarse el botón sin tener seleccionada ninguna entrevista de la lista de emisiones pendientes. Si existe alguna seleccionada se cuenta con el botón *Ninguna* para cancelar la selección. En el caso de una entrevista nueva, la especialidad asociada depende de la seleccionada en la interfaz.

De acuerdo a la especialidad que se encuentre seleccionada, se desplegarán los metadatos correspondientes a la portada de la entrevista. El usuario puede ingresar los datos de los que tiene información. Luego, al aceptar o cancelar los cambios se vuelve al panel inicial de la pestaña *Emitir*.

Una entrevista nueva tiene el estado *Definir Portada*. Esto indica que podrá seguir definiéndose la portada o se podrá decidir finalizar con esta definición y agendar la entrevista.

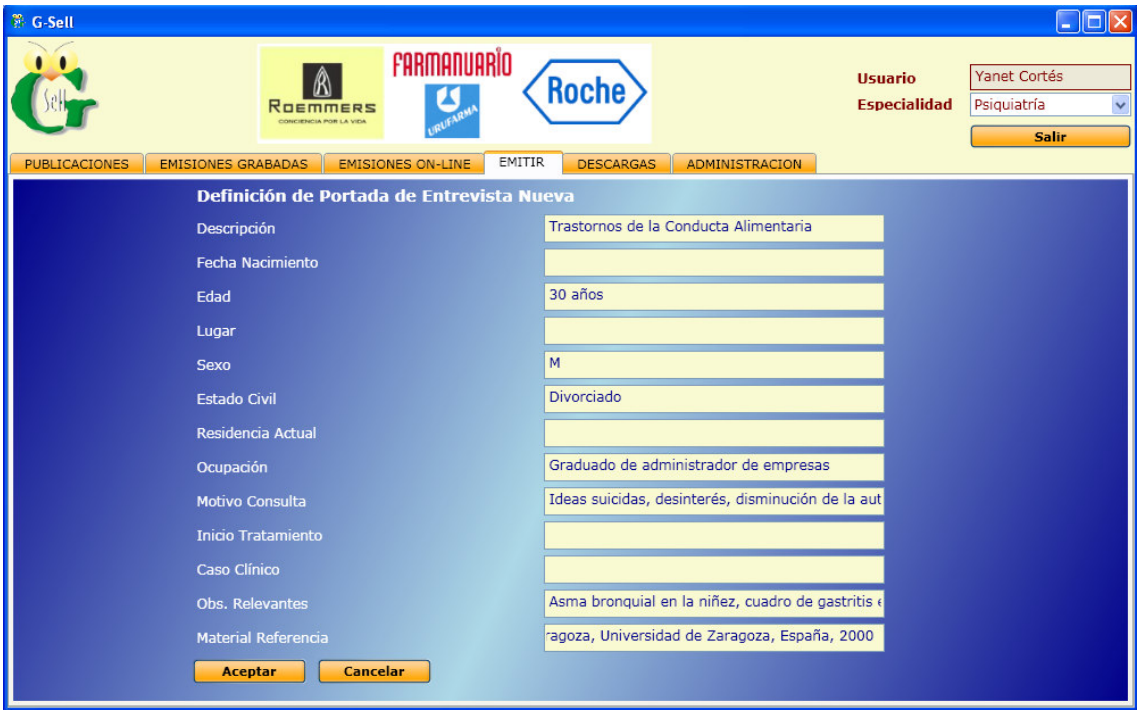


Figura 13: Portada de Entrevista.

5.2 Modificación de Portada de Entrevista



Para continuar con la definición de la portada de una entrevista existente deberá seleccionar la entrevista en la lista de entrevistas pendientes y utilizar el mismo primer botón. Ahora como se invoca con una entrevista seleccionada, se ejecuta el mismo panel que se usa para definir una nueva entrevista pero en modalidad de modificación.

5.3 Agendar



Luego de que se ha completado la definición de la portada de una entrevista se debe agendar. Para esto se utiliza el segundo botón. El concepto de agendar una entrevista corresponde a fijar la fecha y hora en la cual se realizará la emisión en vivo y quienes son los usuarios invitados a presenciarla.



Figura 14: Agendar Entrevista.

Se podrá indicar en la interfaz los usuarios invitados seleccionándolos en la grilla. Los usuarios del sistema que muestran son los que tienen asociada la misma especialidad que tiene la entrevista. Debe recordarse que solamente los usuarios invitados tienen permisos suficientes para visualizar una entrevista al momento de la emisión en vivo.

Es posible en esta interfaz notificar a los usuarios invitados a una entrevista a través del correo electrónico, con el botón *Enviar Correo*, luego de que se haya fijado el momento de la emisión. Cuando se acepten o cancelen los cambios realizados se vuelve al panel inicial de esta pestaña.

En caso de que se haya agendado la entrevista, pasa a estado *Agendar.* Ya no será posible modificar los datos de la portada. Las funcionalidades que quedan disponibles a partir de ahora son modificar la agenda que ya fue realizada, ya sea cambiando el momento en que se va a realizar la emisión o el grupo de usuarios invitados. También se podrá volver a enviar correos notificando del momento en que se realizará la emisión en vivo.

Se agrega a las funcionalidades disponibles la posibilidad de ejecutar la funcionalidad de emisión en vivo anteriormente pactada. Es responsabilidad del usuario realizar esta operación para la fecha y hora establecida.

5.4 Emitir

El  de tercer botón permite acceder al panel en el cual se resuelve la emisión en vivo una entrevista. Para poder realizar la emisión se deberá tener conectada al equipo una cámara y un micrófono.
Es responsabilidad del usuario llevar a cabo la emisión en la fecha y hora pautada anteriormente.

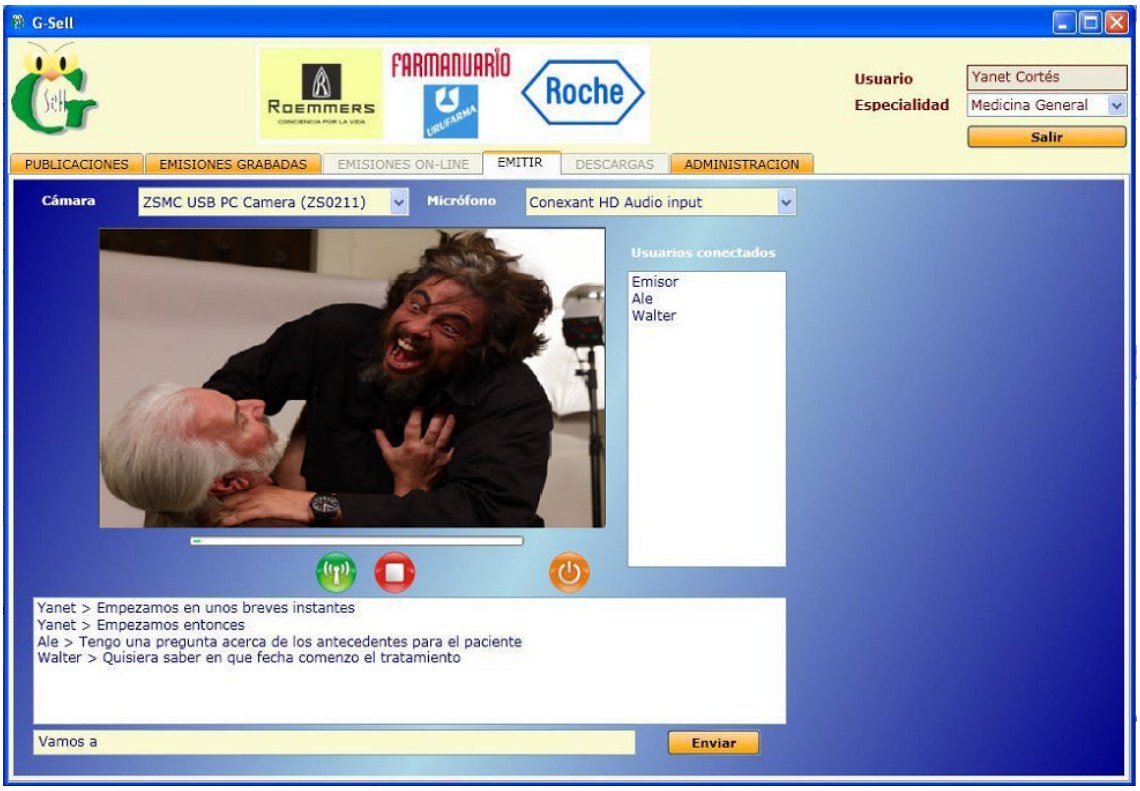


Figura 15: Emitir Entrevista.

Cuando el panel de Emitir se carga ya queda disponible la funcionalidad de intercambio de mensajes. Se puede visualizar una lista de los usuarios conectados a la entrevista.

En la interfaz se debe seleccionar la cámara y el micrófono a usar, en caso de que haya más de uno disponible. Si sólo hay un dispositivo de video o de audio se carga por defecto cuando se carga el panel.

Cuando el usuario quiere enviar un mensaje debe usar el control de la parte inferior. Luego de ingresado el mensaje debe presionar el botón *Enviar*.



Figura 16: Botones de Emisión de Entrevista.

El usuario que emite puede ver la vista previa del video antes de comenzar la emisión. Cuando considere adecuado el comienzo de la emisión del video debe presionar el primer botón, *Comenzar Emisión*.

Para finalizar la emisión del video debe usar el segundo botón, *Detener Emisión*. Luego de esto, aunque la emisión del video ya no se ejecuta más, esta disponible la posibilidad de seguir intercambiando mensajes entre los participantes de la emisión.

Para finalizar toda la emisión debe presionar el botón *Terminar Emisión*. En este momento se finaliza la funcionalidad de envió de mensajes. La entrevista cambia de estado pasando al estado *Emitir* y se vuelve al panel inicial de la pestaña *Emitir*.

5.5 Cerrar

la  Luego de realizada la emisión en vivo de la entrevista, el usuario responsable de misma esta en condiciones de realizar el cierre. Podrá completar la información que considere relevante en la metadata definida para el cierre. Para esto debe utilizar el cuarto botón del panel principal de la pestaña *Emitir*.



Figura 17: Cierre de Entrevista.

Según la especialidad de la entrevista, se desplegaran los metadatos asociados que tenga. De la misma manera en que se definió la portada de la entrevista, se define su cierre. Si se aceptan los cambios la entrevista cambia de estado a *Cerrar*. Cuando se cancela o se aceptan los cambios se vuelve al panel principal de la pestaña *Emitir*.

5.6 Publicar



Cuando se decide publicar una entrevista, se suben al servidor de archivos los asociados a la entrevista. Luego de esto, la entrevista estará disponible para ser consumida por cualquier usuario que tenga permisos sobre ella. Debió haber sido invitado a presenciarla cuando se realizó la emisión en vivo. Para invocar esta funcionalidad debe seleccionarse el quinto botón, *Publicar la Entrevista*.



Figura 18: Publicar Entrevista.

La interfaz permite publicar la entrevista a través del botón *Subir Entrevista*.

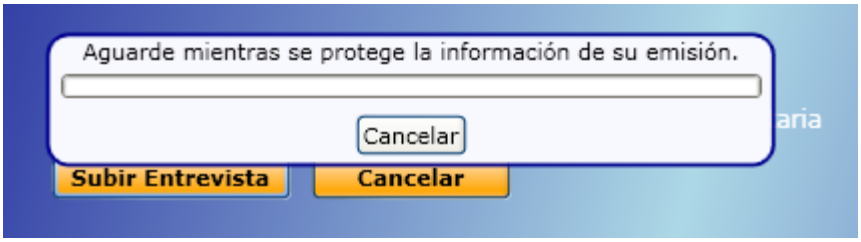


Figura 19: Subir Entrevista.

El usuario deberá esperar mientras se procesa la información a ser enviada desde el equipo local al servidor. Luego podrá volver al panel principal de la pestaña *Emitir* en donde la entrevista recién publicada ya no estará en la lista de *Emisiones Pendientes*.

6. Descargas

En la pestaña de *Descargas* el usuario puede acceder a las emisiones que ha descargado anteriormente. Se encuentran disponibles sin ser filtradas por especialidad.



Figura 20: Pestaña Descargas.

De cada entrevista se puede ver su descripción, el usuario responsable, el tamaño del archivo de la emisión y su largo. Haciendo doble click sobre una de las entrevistas se accede a la entrevista individual.



Figura 21: Entrevista Descargada.

Al igual que en la pestaña de *Descarga*, se visualiza toda la información asociada a la entrevista. Pero en este caso los datos se consumen del mismo equipo, son datos locales.

Con el botón *Ver Emisión* es posible visualizar el video que se encuentra en el disco.



Figura 22: Reproducción de Entrevista Descargada.

En el control ubicado más arriba podrá verse la emisión del video luego de que el usuario la inicie.



Figura 23: Botones de Reproducción de Entrevista Descargada.

Para iniciar la emisión del video debe usarse el primer botón, Comenzar emisión. El segundo botón, *Pausar emisión,* se usa para pausar o reiniciar la emisión. El tercer botón llamado *Detener emisión* es el que detiene la emisión del video. Para salir del panel de emisión de video descargado y volver al panel principal de la pestaña *Descargas* debe presionarse el cuarto botón, *Terminar*.

7. Administración

El panel principal de la pestaña *Administración* esta dividido en dos secciones. En la parte izquierda se cuenta con botones que permiten acceder a las distintas definiciones del sistema. Se pueden visualizar las listas de *Especialidades, Usuarios* o *Metadata* definidas actualmente, del lado derecho.

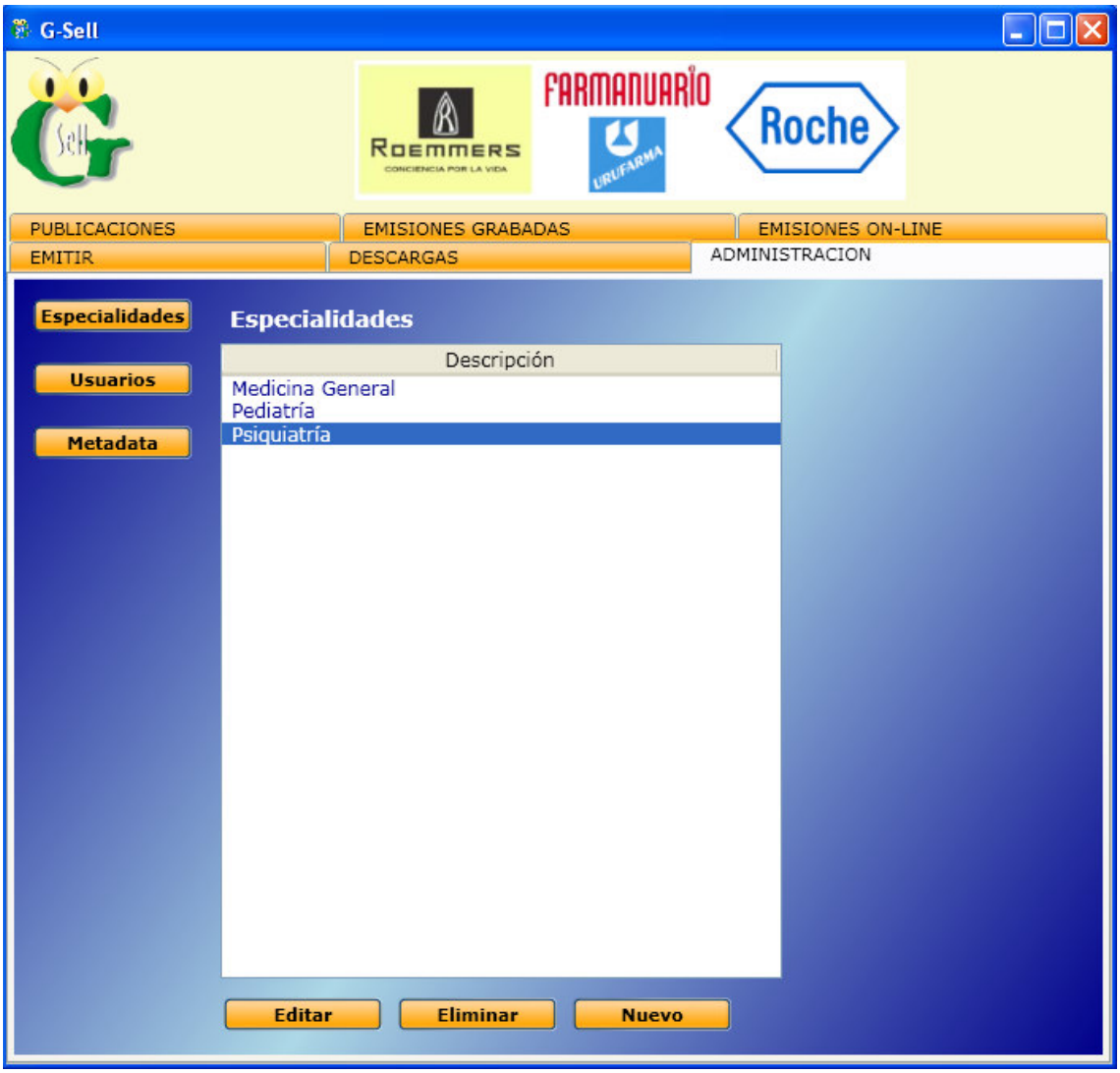


Figura 24: Mantenimiento de Especialidades.

En la lista de especialidades definidas se puede agregar una nueva especialidad, modificar la descripción de una existente o eliminar la especialidad seleccionada.

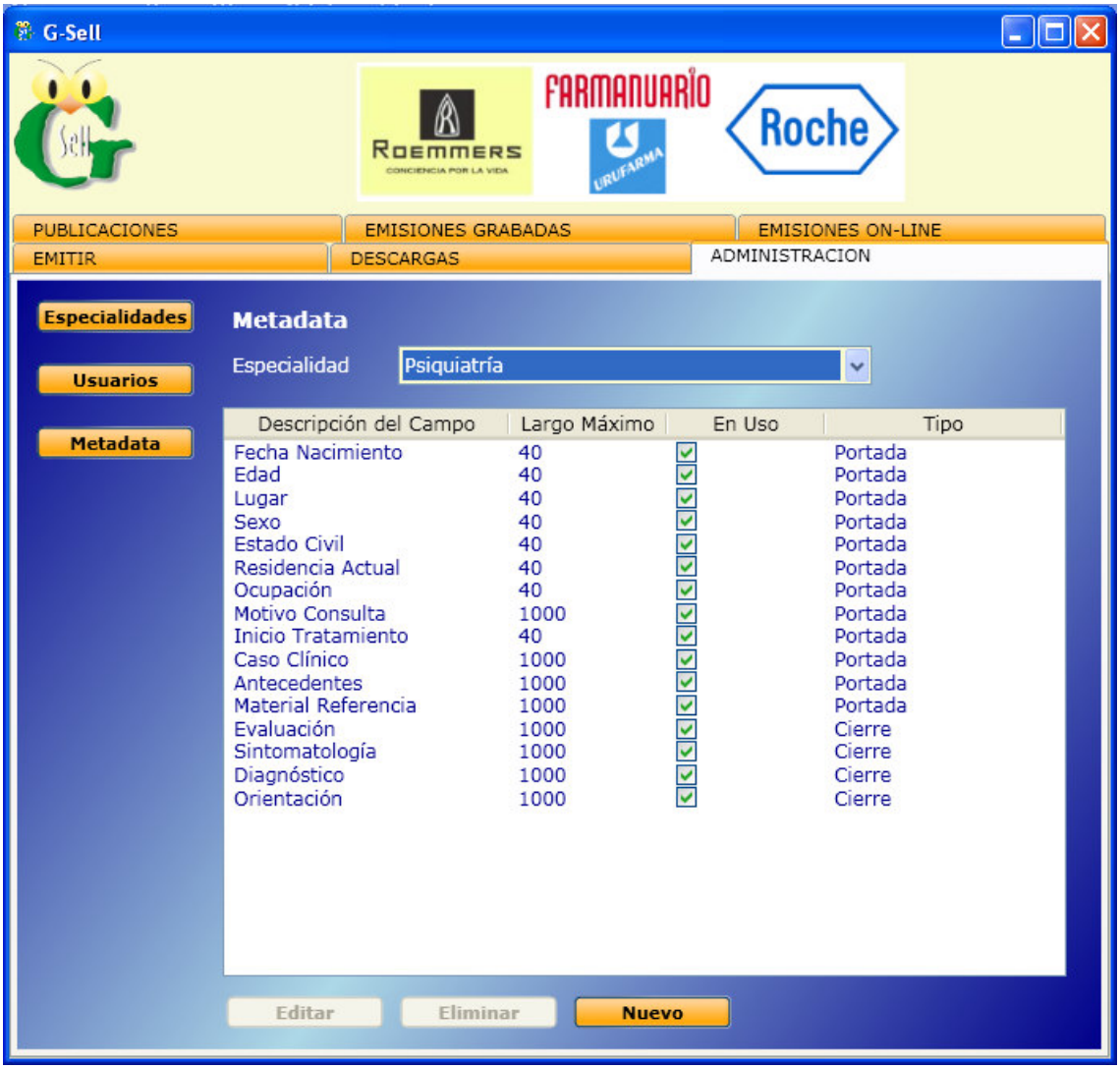


Figura 25: Mantenimiento de Metadata.

En la interfaz donde se visualizan las definiciones de metadata existentes, se debe seleccionar la especialidad a la que los metadatos están asociados. En esta interfaz se puede agregar, modificar o eliminar un metadato.



Figura 26: Mantenimiento de Propiedad de Metadata.

Para una metadata se puede definir la descripción y el largo máximo del campo. El tipo indica si esta asociado a la portada o al cierre de la entrevista. Al aceptar o cancelar se vuelve a la interfaz anterior.

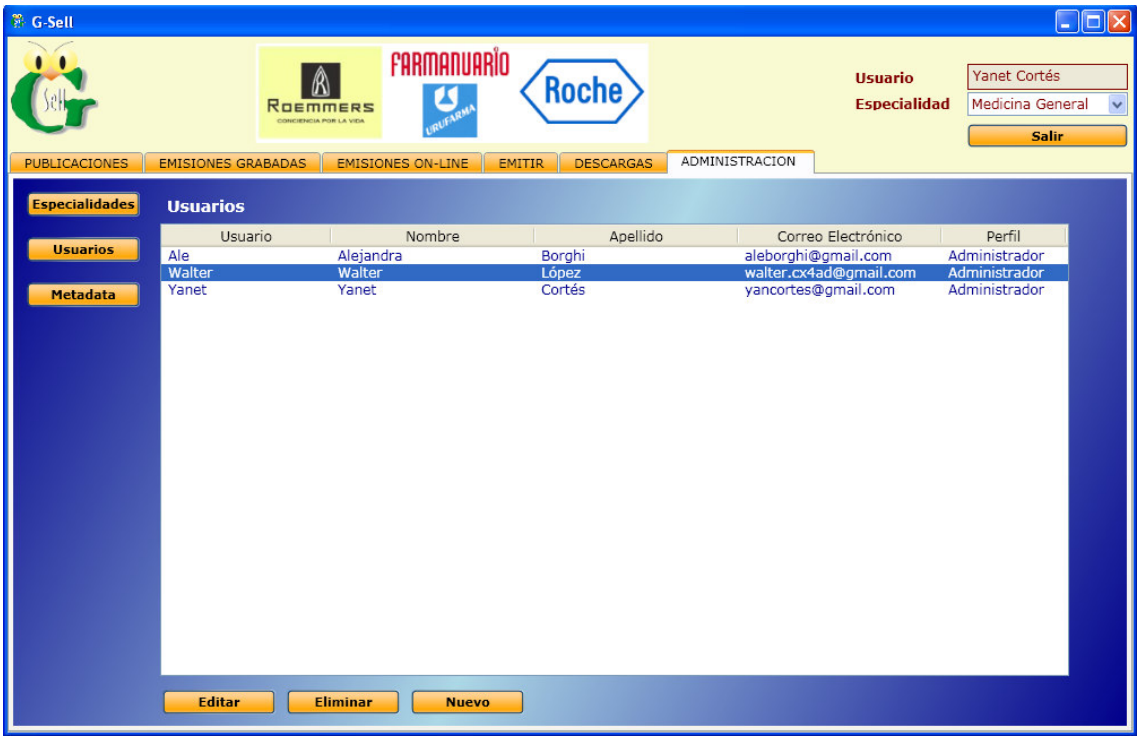


Figura 27: Mantenimiento de Usuarios.

En la lista de usuarios definidos se puede agregar un nuevo usuario, modificar uno existente o eliminar el usuario seleccionado.

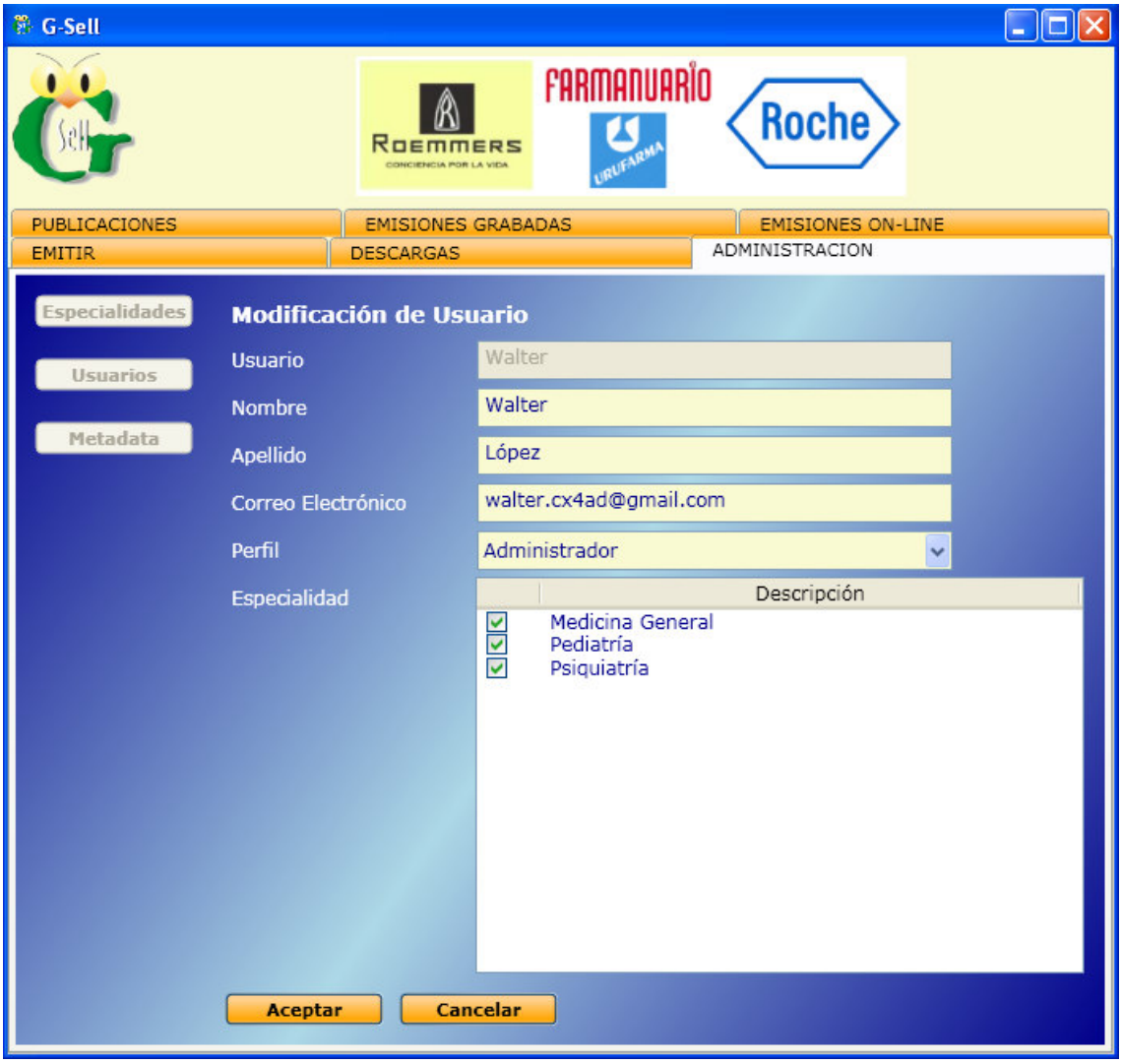


Figura 28: Mantenimiento de Usuario.

Entre otras propiedades se pueden asignar al usuario más de una especialidad asociada. Luego de aceptar o cancelar los cambios se vuelve a la interfaz de usuarios existentes.

Índice de ilustraciones

Figura 1: Pestaña Publicaciones....1
Figura 2: Espacio reservado....1
1. Inicio....2
2. Publicaciones....2
Figura 3: Publicaciones....2
Figura 4: Navegador....3
Figura 5: Alta y modificación de Enlaces....4
Figura 6: Alta y modificación de Publicaciones....4
3. Emisiones Grabadas....5
Figura 7: Emisiones Grabadas....5
Figura 8: Emisión Grabada....6
4. Emisiones On-Line....6
Figura 9: Emisiones On-Line....6
Figura 10: Emisión en vivo....7
Figura 11: Botones de Emisión en Vivo....7
5. Emitir....7
Figura 12: Pestaña Emitir....8
Figura 13: Portada de Entrevista....9
Figura 14: Agendar Entrevista....10
Figura 15: Emitir Entrevista....11
Figura 16: Botones de Emisión de Entrevista....11
Figura 17: Cierre de Entrevista....12
Figura 18: Publicar Entrevista....12
Figura 19: Subir Entrevista....12
6. Descargas....13
Figura 20: Pestaña Descargas....13
Figura 21: Entrevista Descargada....13
Figura 22: Reproducción de Entrevista Descargada....14
Figura 23: Botones de Reproducción de Entrevista Descargada....14
7. Administración....15
Figura 24: Mantenimiento de Especialidades....15
Figura 25: Mantenimiento de Metadata....16
Figura 26: Mantenimiento de Propiedad de Metadata....17
Figura 27: Mantenimiento de Usuarios....17
Figura 28: Mantenimiento de Usuario....18