Universidad de las Ciencias Informáticas

Facultad 2

“Análisis y diseño de un editor de sonido”

Trabajo de Diploma en opción al título de Ingeniero en Ciencias Informáticas

Autora: Yenly Pérez Nuñez

Tutor: Ing. Javier A. León Martínez

Ciudad de La Habana, Cuba, Julio, 2007

“Año 49 de la Revolución”

"Cuando se es joven, se crea. Cuando se es inteligente, se produce. No se adapta, se innova: la medianía copia; la originalidad se atreve."

José Martí.

Agradecimientos

A mami y a papi, por la vida, el amor, el ejemplo, por confiar en mí y darme fuerzas para seguir adelante.

A Yase, que aunque llegue al techo seguirá siendo mi hermanito pequeño y ha sido una de mis fuentes de inspiración para seguir estudiando.

A mima y abuelo Odilo, por todo el cariño y el consentimiento de siempre.

A mis tíos, por la preocupación y el cariño de padre que siempre me han brindado.

A tía María y Marlenis, que han sabido hacer de cada sobrino un hijo y tantas fuerzas me han dado para seguir adelante y no pensar en arrepentirme nunca.

A mi primos Ramy y Yunior que siempre me apoyaron.

A mis amigas Daymí y Yilo que siempre me esperaron con el mismo cariño, sin importar la distancia y el tiempo que estábamos separadas.

A mi tía Gudelia y su familia por acogerme como otro más de la casa.

A Clari, Edelia María y el resto de la familia por quererme como su propia hija y siempre haber estado ahí cuando los necesité.

A mis amigos, los que siempre han estado: Maide, Omar, Jorgito, Yoisel, Lia, Raúl, Mañi, Laura, y los nuevos que no por nuevos dejan de ser importantes como: Hany, la mama, Anniel, Misly, Carlitos y Adonis que han llegado para quedarse.

A mis compañeros, por las risas, las ocurrencias y los años compartidos.

A la revolución y a Fidel por darnos la oportunidad de estudiar en esta maravillosa Universidad.

A mis compañeros de proyecto, por la paciencia que han tenido conmigo, principalmente a Vladimir, a Yunior y al Enano que tantas veces levanté de la PC.

A mi tutor por darme fuerzas para seguir adelante y por confiar en mí, además de la paciencia que me ha tenido.

A Yoisel y a Alamino por estar siempre ahí, dispuestos a aclararme mis dudas, sin poner mala cara.

A la profe Lourdes por su ayuda.

A mi oponente por sus oportunas críticas.

A mis padres,

familia y

a Clari.

DECLARACIÓN DE AUTORÍA

Declaro ser autora de la presente tesis y reconozco a la Universidad de las Ciencias Informáticas los derechos patrimoniales de la misma, con carácter exclusivo.

Para que así conste firmo la presente a los __ días del mes de _____ del año 2007.

Yenly Pérez Nuñez **Ing. Javier A. León Martínez**

_____ _____

Firma del Autor Firma del Tutor

Resumen

El propósito de este trabajo consiste en la realización del análisis y diseño de un software de producción nacional para realizar ediciones profesionales de audio en la radio cubana que cumpla con los siguientes requisitos: debe ser capaz de conectarse a una base de datos para salvar las ediciones o cargar ficheros de audio, además de ser multipista y de fácil manejo.

Para lograr esto se entrevistó al cliente para capturar los requisitos funcionales y no funcionales, que debía tener el software. Se realizó el modelo del dominio para entender el negocio, se plantearon las reglas del negocio, además del diagrama de casos de uso del sistema, se describieron los casos de uso del sistema. En la fase de análisis y diseño se desarrollaron los diagramas de clases, tanto del análisis como del diseño, además se realizaron los diagramas de secuencia del diseño. Se definieron los patrones que se iban a utilizar.

INDICE

Introducción

El desarrollo de la informática a ha llevado aparejado un desarrollo de los software para la edición de sonido, esto a permitido que se desarrolle una competencia en el mundo de la edición, existen editores muy sencillos que permiten que los aficionados a la música, puedan crear su propia música, además hay un disímil número de editores profesionales, pero estos necesitan de mayores recursos de hardware, para un funcionamiento eficiente.

¿Qué es un editor de sonido?
Un editor de sonido es un software que se utiliza para mezclar músicas, cortar, pegar, eliminar, añadir efectos a pistas, existen editores muy sencillos que se pueden utilizar en el hogar, para hacer ediciones no profesionales y otros para usos profesionales que pueden ser multipista.

Problema a resolver:

No existe un editor de sonido de factura nacional que cumpla con las siguientes características: que sea multipista y que permita conectarse a la base de datos para salvar las ediciones o cargar el fichero.

Actualidad y necesidad del trabajo:

En la actualidad en la radio cubana se trabaja con los editores de sonido

- X-Track
- Cool Edit
- Sound Forge

Estos editores no permiten conectarse a la base de datos, por lo que debemos ser capaces de resolver ese problema, para que sea más fácil y eficiente el trabajo de ediciones en la radio cubana.

Antecedentes:

Características de Sound Forge

Sound Forge está reconocido como un estándar para la edición de audio en la plataforma Windows. Contiene una gran variedad de opciones para el procesamiento de audio. Soporta video para Windows, lo que le permite sincronizar audio y video con la precisión de un fotograma. Soporta además una gran lista de formatos de audio, incluyendo: RealAudio, RealVideo, formato de ASF, y Java. También soporta plug-ins basados en la arquitectura de servicios de DirectX.

Algunas de sus características más destacadas son: edición no lineal en el disco duro; variedad de efectos de audio, procesos, y herramientas; lee y escribe los formatos de todos los ficheros soportados; producción con calidad de estudio para profesionales; compresión de ficheros en 8 bits para su distribución; listas de reproducción y listas de regiones para masterizado de CD; soporte de filtros especiales para la reducción de ruido, etc. [1]

Características de Audacity

Es el editor de audio de software libre por excelencia, dado que posee una Licencia Publica General de GNU (GNU GPL). Puede ser utilizado sobre diferentes plataformas (Mac OS, Linux, Windows). Puede trabajar con diferentes formatos de audio (WAV, AIFF, AU, Ogg Vorbis y diferentes versiones de MPEG), aunque no trabaja con los formatos WMA, AAC, o el resto de los formatos de archivo propietarios o restringidos. Graba y edita muestras de 16-bit, 24-bit y 32-bit, hasta una frecuencia de muestreo máxima de 96 KHz. Las frecuencias de muestreo y formatos son convertidos mediante un proceso de alta calidad. Permite mezclar pistas con diferentes frecuencias de muestreo o formatos. [2]

Características de Cool Edit:

Cool Edit incluye un número importante de efectos, tales como: amplificación, *fading*, inversión, normalización, silencio, vibración, eco, coro, entre otros. También dispone de las funciones características de edición: cortar, copiar, pegar, insertar, mezclar.

Además de editar ficheros de audio, permite crear nuevos, por ejemplo grabándolos con un micrófono desde el propio programa, o extrayéndolos de un CD de audio con una herramienta también integrada en Audio Editor Pro. Soporta los formatos de audio más comunes (WAV, MP3, WMA, RAW, VOX, CDA, Ogg Vorbis). Puede realizar conversiones entre varios de ellos, y permite también visualizar y editar la información contenida en los tags de los ficheros. [3]

Características de Fleximusic Wave Editor

Este editor de audio te permite crear, editar, grabar, añadir efectos especiales y reproducir ficheros de audio en varios formatos, como son WAV, MP3, RAW, AU y SND.

Permite realizar operaciones básicas de edición como cortar, copiar y pegar, así como añadir efectos y filtros como eco, ruido, silencio, reproducción inversa, mezcla de canales, etc. [4]

Características de X-Track.

Este software editor de audio digital mutipista es intuitivo, completo y flexible para editar emisiones, la post-producción de videos, creación de CDs y otras aplicaciones.

Presenta un número ilimitado de pistas de audio virtual que son dinámicamente asignadas a las entradas y las salidas, con pistas sin asignar que sirven de pistas de trabajo. Sencillo de manejar, con edición no destructiva y funciones del proceso que incluyen copiar, insertar, remplazar, "drag and drop", rellenar el patrón, ajustar un marcador, acortamiento del tiempo, mezcla de pistas. Las funciones de auto localización del video y audio permiten marcar y llamar rápidamente los puntos de edición.

Una característica clave de X-Track es su capacidad para crear simples archivos de sonido en el formato PCM o MPEG desde una mezcla multipista, que incluye ediciones en niveles de automatización y niveles de pistas. [5]

Como se dijo anteriormente, en la radio cubana los editores de audio más utilizados son: Cool Edit, Sound Forge y X-Track, a continuación le mostramos un resumen con las ventajas y desventajas más importantes de cada uno de ellos.

Positivo de X-Track

- Opera sobre las tarjetas Digigram.
- Trabaja con ficheros comprimidos, gracias a las tarjetas Digigram.
- Permite lograr bajar el volumen entre dos puntos de manera muy sencilla.
- Es multipista.

Negativo de X-Track

- Es lento en el procesamiento.

Positivo de Sound Forge

- Versátil para aplicaciones sencillas.

Negativo de Sound Forge

- No es multipista.

Positivo de Cool Edit

- Es multipista.
- Gran rapidez en el procesamiento.

Negativo de Cool Edit

- No tiene el control que posee X-Track para bajar el volumen entre dos puntos, por lo que tiene que trabajar con los gráficos.

Objetivo de la investigación

Analizar y diseñar una aplicación cliente-servidor para la edición de audio, en la radio cubana.

Campo de acción

El campo de acción radica en editar sonido en la radio cubana.

Tareas a cumplir por el estudiante

- Estudio del procesamiento de audio.
- Estudio de los protocolos de comunicación de .NET con Postgre.
- Estudio de la arquitectura, acorde a las necesidades de las ediciones.
- Investigar el estado del arte.
- Estudiar herramientas y tecnologías.
- Desarrollar el Modelo del Negocio y la captura de Requisitos.
- Desarrollar el Modelo de Análisis y Diseño.
- Entrevistar al cliente.

El documento esta estructurado en 4 capítulos:

- Capítulo I denominado “Fundamentación teórica”. Se analizan algunas de las herramientas y lenguajes de programación más utilizadas en el mundo para el desarrollo de las aplicaciones, y se plantea la metodología a seguir en la elaboración del mismo.

- Capítulo II denominado “Características del Sistema”. Se definen las reglas del negocio y los conceptos fundamentales del sistema, se determinan además los requerimientos funcionales y no funcionales del mismo, agrupándolos en casos de usos. Se da una idea general de la concepción del sistema.
- Capítulo III denominado “Análisis y Diseño del Sistema”. Se determinan las clases del análisis y del diseño que se utilizarán en el sistema y la relación entre ellas. Además se muestra todo el proceso de obtención de la base de datos y se definen los patrones de diseño.
- Capítulo IV denominado “Estudio de factibilidad”. Se calcula el costo del proyecto y el tiempo de duración del mismo.

Capítulo I. Fundamentación teórica.

1.1 Introducción

A lo largo del desarrollo de todo software es preciso realizar una serie de estudios que garanticen el avance del mismo, en búsqueda de cumplir los requerimientos planteados por el usuario. Se realiza un análisis de las tecnologías actuales de la informática y las comunicaciones consideradas a la hora de implementar el presente software.

1.2 Metodología

Las tendencias actuales predisponen la existencia de un proceso bien definido y bien gestionado para poder lograr cumplir las exigencias requeridas en todo producto informático.

El **proceso de desarrollo de software** "*es aquel en que las necesidades del usuario son traducidas en requerimientos de software, estos requerimientos transformados en diseño y el diseño implementado en código, el código es probado, documentado y certificado para su uso operativo". Concretamente "define quién está haciendo qué, cuándo hacerlo y cómo alcanzar un cierto objetivo. [6]*

En concreto se traduce en el conjunto de actividades necesarias para transformar los requisitos de un usuario en un sistema de software:

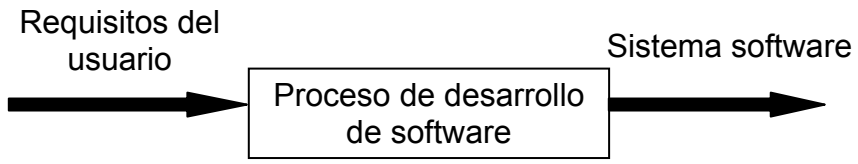


Figura 1. Proceso de desarrollo de software.

Existen varias metodologías para controlar el proceso para el desarrollo de software, una propuesta es el Proceso Unificado de Desarrollo de Software (Rational Unified Process, RUP), resultado de la evolución e integración de diferentes metodologías de desarrollo de software. RUP está basado en componentes, lo cuál quiere decir que el sistema en construcción está formado por componentes de software interconectados a través de interfaces bien definidas. Utiliza el *Lenguaje Unificado de Modelado* (Unified Modeling Languaje, UML) para preparar todos los esquemas, garantiza la elaboración de todas las fases de un producto de software orientado a objetos.

El UML es un lenguaje que permite modelar, construir y documentar los elementos que forman un sistema de software orientado a objetos. Entrega una forma de modelar conceptos como lo son procesos de negocio y funciones de sistema, escribir clases en un lenguaje determinado, esquemas de bases de datos y componentes de software, entre otros. [7]

1.2.1 Proceso Unificado de Desarrollo de Software (RUP)

Los principales rasgos que definen unívocamente al Proceso Unificado de Desarrollo de Software lo constituyen el ser dirigido por casos de uso, centrado en la arquitectura así como iterativo e incremental.

RUP está **dirigido por casos de uso,** el proceso de desarrollo sigue una trayectoria que avanza a través de los flujos de trabajo generados por los casos de uso. Los casos de uso se especifican y diseñan en el principio de cada iteración, y son la fuente a partir de la cual los ingenieros de prueba construyen sus casos de prueba. Los casos de uso describen la funcionalidad total del sistema, pensada en términos de la importancia de la misma para el usuario. Esto no significa que se desarrollen aisladamente respecto a la arquitectura, sino que se desarrollan a la vez, madurando ambos según avanza el ciclo de desarrollo. Los casos de uso guían a la arquitectura del sistema (como parte del proceso) y ésta influye en la selección de los casos de uso. La arquitectura involucra los elementos más significativos del sistema y está influenciada entre otros por las plataformas software, los sistemas operativos, los sistemas de gestión de bases de datos, además de otros como sistemas heredados y requerimientos no funcionales.

Por esta razón RUP está **centrado en la arquitectura**, lo que invoca más la relación con los principios de la usabilidad. Desde que se planteó por primera vez el modelo **incremental** de desarrollo de software, y se establecieron sus ventajas con respecto al modelo de cascada, siempre se ha recomendado dividir los proyectos en pequeños ciclos o **iteraciones** a través de cada una de las fases por las que pase. RUP divide el proceso de desarrollo de software en cuatro fases, (inicio, elaboración, construcción y transición), dentro de las cuales se realizan varias iteraciones en número variable, según el proyecto, las cuales se definen de acuerdo al nivel de madurez que alcanza el producto que se van obteniendo con cada actividad ejecutada. La terminación de cada fase ocurre en el hito correspondiente a cada una, donde se evalúa que se hayan cumplido los objetivos de la fase en cuestión. Y con la terminación de la fase de inicio se puede ya determinar la factibilidad tanto operativa como económica del proyecto, lo cuál nos lleva a tomar la decisión de continuarlo o no realizarlo. [8]

1.2.2 El Lenguaje de Modelado Unificado (UML)

Es un lenguaje de propósito general para el modelado orientado a objetos. UML es también un lenguaje de modelación visual que permite una abstracción del sistema y sus componentes, se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software. Nos permite entender, diseñar, configurar, mantener y controlar la información sobre los sistemas a construir.

UML no es un lenguaje de programación. Las herramientas de modelación pueden ofrecer generadores de código a partir de especificaciones UML para una gran variedad de lenguaje de programación, así como construir modelos por ingeniería inversa a partir de programas existentes.

Permite la modificación de todos sus miembros mediante estereotipos y restricciones. Un estereotipo nos permite indicar especificaciones del lenguaje al que se refiere el diagrama de UML. Una restricción identifica un comportamiento forzado de una clase o relación, es decir mediante la restricción estamos forzando el comportamiento que debe tener el objeto al que se le aplica.

UML es un estándar, su utilización es independiente del lenguaje de programación y de las características de los proyectos, ya que ha sido diseñado para modelar cualquier tipo de proyectos. [8]

1.2.3 Rational Rose Enterprise

Rational Rose Enterprise es un entorno de modelado que permite generar código a partir de modelos Ada, ANSI C++, C++, CORBA, Java/J2EE, Visual C++ y Visual Basic. Al igual que todos los productos de Rational Rose, ofrece un lenguaje de modelado común que agiliza la creación del software. [9]

Funciones:

- Soporte a modelos de análisis, ANSI C++, Rose J y Visual C++
- Soporte para compilación y descompilación de las construcciones más habituales de Java 1.5.
- Generación de código en lenguaje Ada, ANSI C++, C++, CORBA, Java y Visual Basic, con funciones configurables de sincronización entre los modelos y el código.
- Soporte para Enterprise Java Beans 2.0.
- Funciones de análisis de calidad de código.
- Complemento de modelado Web que incluye funciones de visualización, modelado y herramientas para desarrollar aplicaciones Web.
- Modelado en UML para diseñar bases de datos, que integra los requisitos de datos y aplicaciones mediante diseños lógicos y analíticos.
- Creación de definiciones de tipo de documento DTD en XML.
- Integración con otras herramientas de desarrollo de IBM Rational.
- Posibilidad de publicar en las Web modelos e informes para mejorar la comunicación entre los miembros del equipo.

1.3 Sistemas Gestores de Bases de Datos (SGBD)

En 1964 se conciben los primeros SGBD, con estos se pretendió dar un viraje a los Sistemas de Archivos, los cuales se limitaban a la estructuración del almacenamiento físico de los datos. Con los SGBD se logró, por medio de actividades integradas, verlos físicamente en un solo almacenamiento, pero lógicamente se manipulan a través de esquemas compuestos por estructuras donde se establecen vínculos de integridad, métodos de acceso y organización física sobre los mismos, permitiendo así obtener valores agregados de utilización.

"*En 1970 el Dr. Edgar F. Codd propuso el Modelo de Datos Relacional, este modelo es el que ha marcado la línea de investigación por muchos años, representa al mundo real mediante tablas relacionadas entre sí por columnas comunes. Viendo la necesidad de mejorar este estándar se desarrollaron los Sistemas Gestores de Base de Datos Relacionales (SGBDR) cuyas características hacen al sistema mucho más eficiente que los sistemas de manejos de archivos"* [10]

Un SGBDR es un conjunto de datos relacionados entre sí y un grupo de programas para tener acceso a esos datos, permitiendo concurrencia y recuperación. La persistencia de un SGBDR hace referencia a la conservación de los datos después de la finalización del proceso que los creó, y la concurrencia se refiere a la capacidad del sistema para gestionar a múltiples usuarios interactuando al mismo tiempo sobre el mismo. Entre los SGBDR más conocidos se encuentran SQL Server, Oracle y PostgreSQL a continuación se dará una breve explicación de este último.

1.3.1 PostgreSQL

PostgreSQL es un motor de base de datos, es servidor de **base de datos relacional** libre, liberado bajo la licencia BSD. Es una alternativa a otros sistemas de bases de datos de código abierto (como MySQL, Firebird y MaxDB), así como sistemas propietarios como Oracle o DB2.

Características PostgreSQL:

Las funciones permiten subir bloques de código que se ejecuten en el servidor. Estas funciones pueden escribirse en una variedad de lenguajes, algunos de los más importantes son PL/pgSQL, C, C++ y Java.

Puede definirse si las funciones serán ejecutadas con los permisos del llamador o del usuario que definió la función.

Mediante un sistema denominado MVCC (Acceso concurrente multiversión) PostgreSQL permite que mientras un proceso escribe en una tabla, otros accedan a la misma tabla sin necesidad de bloqueos. Cada usuario obtiene una visión consistente de lo último a lo que se le hizo *commit*. Esta estrategia es superior al uso de bloqueos por tabla o por filas común en otras bases, eliminando la necesidad del uso de bloqueos explícitos. [11]

1.4 Tecnología .NET

1.4.1 Microsoft.NET

Microsoft.NET es el conjunto de nuevas tecnologías en las que Microsoft ha estado trabajando durante los últimos años con el objetivo de obtener una plataforma sencilla y potente para distribuir el software, de manera que los lenguajes de programación y modelo de componentes con los que hayan sido desarrollados puedan comunicarse y combinarse sin importar la plataforma, a ella se le denominó plataforma .NET.

Microsoft, para crear aplicaciones escalables y distribuidas ha publicado el Framework .NET SDK, que incluye las herramientas necesarias tanto para el desarrollo de aplicaciones como para su distribución y ejecución. Visual Studio.NET es la herramienta por excelencia que permite el desarrollo de aplicaciones desde una interfaz visual basada en ventanas.

1.4.2 Framework .NET

Se le llama Framework, ("entorno de trabajo"), a las Bibliotecas de Clase Base, (también llamadas BCL) y el Common Language Runtime, (CLR) que resulta ser el corazón de la plataforma .NET y ofrece una interoperatividad multi-lenguaje, o sea, la característica que códigos de un lenguaje, para una aplicación, pueda utilizar códigos de otro lenguaje sin inconvenientes. [12]

1.4.3 Visual Studio .NET 2005

Es un entorno integrado de programación (llamado en inglés por siglas: IDE) para sistemas Windows. Se soportan varios lenguajes de programación -oficialmente Visual C++, Visual C#, Visual J#, ASP.NET y Visual Basic .NET.[13]

Visual Studio permite a los desarrolladores crear aplicaciones, sitios y aplicaciones web, así cómo servicios web en cualquier entorno soportado por la plataforma .NET (a partir de la versión 6). Así se pueden crear aplicaciones que se intercomuniquen entre estaciones de trabajo, páginas web y dispositivos móviles. Es una plataforma de ejecución intermedia multilenguaje. Los programas desarrollados en .NET no se compilan en lenguaje máquina (cómo C++), sino que se compilan en un lenguaje intermedio (CIL - Common Intermediate Language). El lenguaje usado es Microsoft Intermediate Language (MSIL). Incluye tipos genéricos, similares en muchos aspectos a las plantillas de C++. Con esto se consigue encontrar muchos más errores en la compilación en vez de en tiempo de ejecución, incitando a usar comprobaciones estrictas en áreas donde antes no era posible. Incluye un diseñador de implantación, que permite que el diseño de la aplicación sea validado antes de su implantación. También se incluye un entorno para publicación web y tests de carga para comprobar el rendimiento de los programas bajo varias condiciones de carga.

1.4.4 SharpDevelop

SharpDevelop es un entorno de desarrollo integrado para la plataforma .NET. Soporta las versiones del FrameWork de Microsoft y de Ximian Soporta desarrollo de interfaces, clases, namespaces y proyectos en C# y VB.NET, además de permitir importar los proyectos creados con Microsoft Visual Studio .NET [14]

- Completado de código. Soporta el uso de la combinación de teclas Ctrl + Espacio.
- Depurador incorporado.
- Herramientas para "Ir a Definición", "Encontrar referencias" y "renombrado".
- Títulos para títulos y para depuración.
- Conversor bidireccional entre C# y Visual Basic .NET, y unidireccional hacia Boo.
- Escrito enteramente en C#.
- Compilación de código directamente dentro del entorno de desarrollo integrado.
- Complementos para ILAsm y C++.
- Integración con herramientas de pruebas unitarias NUnit y MbUnit.
- Analizador para ensamblado FxCop.
- Previsualización de documentación XML.
- Gran integración con plantillas a la hora de añadir o crear ficheros, proyectos o compiladores.
- Escritura de código C#, ASP.NET, ADO.NET, XML y HTML.
- Coloreado de sintaxis para los lenguajes C#, HTML, ASP, ASP.NET, VBScript, Visual Basic .NET, y XML.
- Llaves inteligentes en la escritura de código.
- Gestión de marcadores (favoritos).
- Soporte para plantillas de código.

1.5 Lenguajes

1.5.1 C#, (o Visual C#)

Este es el nuevo lenguaje inspirado en C/C++, similar a Java. Este lenguaje tiene una sintaxis basada en C/C++ con una estructura similar a Java pero con características especiales que lo hacen muy estructurado, sencillo, poderoso y de alto nivel. Hace uso de las BCL lo cual sirve para cualquier otro lenguaje .NET. [16]

Ofrece una serie de ventajas basadas fundamentalmente en aspectos tales como:

- Sencillez: Es un lenguaje simple y de fácil aprendizaje,
- Soporta el trabajo con punteros manteniéndolos en espacio de código que el usuario declara como unsafe (inseguro).
- Orientado a Objetos: Es un lenguaje diseñado para la implementación de software orientado a objetos. Implementa conceptos como la herencia, el tratamiento de estructuras, la abstracción, la herencia, el polimorfismo, la encapsulación, entre otros.
- Modernidad: Es un lenguaje joven, surgido en el año 2000 y muy similar al Java.
- Distribuido: Está concebido para trabajar en un entorno conectado en red. Cuenta con una amplia biblioteca de clases para comunicarse mediante TCP/IP: HTTP, FTP, etc.
- Seguridad de tipos.
- Instrucciones seguras.
- Sistema de tipos unificado.
- Extensibilidad de tipos básicos.
- Orientación a componentes.
- Extensibilidad de operadores.
- Extensibilidad de modificadores.
- Versionabilidad.
- Eficiencia.
- Compatibilidad.

1.6 Conclusiones

En este capítulo se realizó un análisis de la tecnología a utilizar, de la herramienta para modelar el problema, los principales lenguajes de programación y plataformas de desarrollo.

Después de realizado todo el proceso de investigación se decidió que el Sistema se implementará utilizando la plataforma .NET, el lenguaje C#, metodología RUP utilizando UML como lenguaje de modelación y como gestor de base de datos PostgreSQL.

Capítulo 2. Características del sistema

2.1 Introducción

En este capítulo se realiza el análisis del proceso de desarrollo del software. Este proceso de desarrollo se centró en la metodología RUP, haciendo uso del UML para la modelación de los artefactos. Como herramienta CASE para asistir el proceso de desarrollo fue utilizado el Rational Rose.
Se presenta el modelo de dominio, como alternativa al modelo de negocio, para entender el contexto en que se ubica el sistema.
Se presentan los requerimientos funcionales y no funcionales especificados por el usuario. Se describen los casos de uso y la propuesta del sistema a desarrollar.

2.2 Situación Problémica

En el ICRT se trabaja la edición de sonido con un software (X-Track) multipista, que está diseñado para operar sobre las tarjetas de audio profesionales de la firma francesa Digigram, el cual presenta algunos inconvenientes que hacen crítica su utilización:

- Si se trabaja con más de 2 pistas puede no funcionar de manera satisfactoria.
- En ocasiones no permite salvar la edición.
- El codificador no funciona si el usuario no es administrador.
- Las licencias son caras.
- No hay estabilidad en las ediciones muy grandes.
- Necesita una llave de hardware que es específica y cara.
- No permite conectarse a la base de datos para guardar y/o cargar las ediciones.

2.3 Modelo del dominio

Durante el análisis de los procesos surgieron varios conceptos relacionados entre si, estos definen el dominio del sistema, que permite a los usuarios, clientes, desarrolladores e interesados, utilizar un vocabulario común para poder entender el contexto en que se ubica el mismo, para capturar correctamente los requisitos y lograr una solución adecuada.

2.3.1 Conceptos del Modelo del Dominio

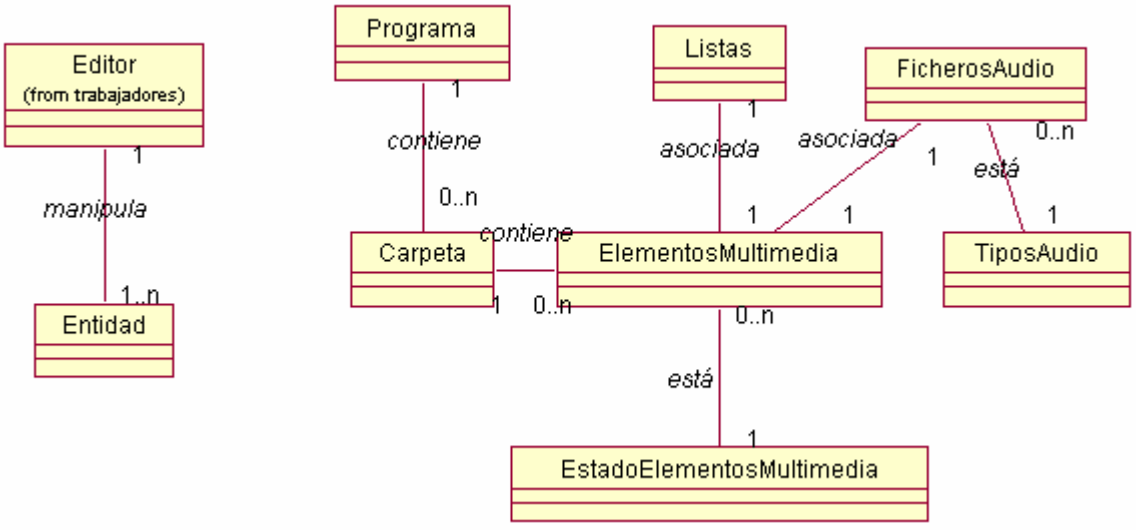
Editor: Es el trabajador encargado de realizar las ediciones, actualizar la BD.

Entidad: Representa las entidades que intervienen en el sistema.

Programa, Carpeta, ElementosMultidedia, FicherosAudio, Listas, EstadoElementosMultimedia, TiposAudio: Instancias de Entidad.

A continuación se muestra como interactúan los conceptos manipulados en el sistema:

Diagrama 1. Modelo del dominio.



2.3.3 Descripción del modelo del dominio

El editor es el encargado de actualizar la base de datos y para ello manipula las entidades. Programa contiene ninguna o muchas carpetas, una carpeta contiene ninguno o muchos elementos de multimedia, un elemento de multimedia solo puede tener un estado, un estado de elementos de multimedia, puede estar en ninguno o muchos elementos de multimedia, un elemento de multimedia está asociado a una lista o a un fichero de audio, un tipo de audio puede estar en ninguno o muchos ficheros de audio.

2.4 Reglas del negocio

El sistema deberá trabajar con el formato .wav, por ello al cargar el fichero lo convierte a esta extensión.

2.5 Propuesta del sistema

Se desarrollará el análisis y diseño de una aplicación cliente - servidor para la radio cubana, y posteriormente implementar un software que permita: cargar las pistas de la base de datos, guardar las ediciones en la base de datos. Esta aplicación se diferenciará del X-track, en que va permitir la conexión con la base de datos y del Sound Forge en que este va a ser un editor multipista.

2.6 Especificación de los requerimientos

2.6.1 Requerimientos funcionales

Como punto de partida para determinar que debe lograr el sistema, se han identificado los siguientes requerimientos funcionales que representan las condiciones y capacidades que el software debe cumplir:

Autenticar usuario

1. Autenticar usuario

1.1 Introducir nombre de usuario y contraseña

1.2 Validar datos introducidos

Añadir efectos

2. Añadir efectos

2.1 Reverse

2.2 Eliminar silencio

2.3 Insertar silencio

2.4 Modificar ganancia

2.5 Silenciar

2.6 Puerta de ruido (noise gate)

2.7 Normalizar

2.8 Aplicar envolvente

2.9 Modular la amplitud

2.10 Modificar la frecuencia de muestreo

2.11 Resemple (remuestreo)

2.12 Transponer

Permitir Procesamiento Multipista

3. Permitir Procesamiento Multipista

3.1 Abrir varios ficheros

Gestionar TipoAudio

4. Gestionar TipoAudio

4.1 Insertar

4.2 Eliminar

4.3 Modificar

Gestionar Carpeta

5. Gestionar Carpeta

5.1 Insertar

5.2 Eliminar

5.3 Modificar

Gestionar Lista

6. Gestionar Lista

6.1 Insertar

6.2 Eliminar

6.3 Modificar

Gestionar Programa

7. Gestionar Programa

7.1 Insertar

7.2 Eliminar

7.3 Modificar

Gestionar Elemento

8. Gestionar Elemento

8.1 Insertar

8.2 Eliminar

8.3 Modificar

2.6.2 Requerimientos no funcionales

Lograr cumplir las expectativas del usuario, implica, además de lograr que el software funcione, que lo haga de la mejor manera posible en base a los requisitos no funcionales que especifique el mismo; el presente trabajo se centró en:

Portabilidad: El sistema debe ser multiplataforma.

Apariencia o interfaz externa: El producto debe tener un diseño sencillo, permitiendo la utilización del sistema sin necesidad de mucho conocimiento informático.

Zoom visualización

Para trabajar los fragmentos con mayor detalle es conveniente poder modificar la escala de visualización. El zoom se puede aplicar a dos parámetros: el tiempo (eje horizontal) y la amplitud (eje vertical).

Características del sistema.

Software: Para el adecuado funcionamiento del sistema, la computadora debe tener instalado alguno de los sistemas operativos y los programas que a continuación se muestran:

Tabla 1**:** Requisitos de software para el cliente.

Software	**Requisitos**
Sistema Operativo	Windows XP Professional
Plataforma	SharpDevelop 2.0.1.1710 Microsoft. NET Framework SDK V2.0

Tabla 2**:** Requisitos de software para el servidor.

Software	**Requisitos**
Sistema Operativo	Windows Server 2003
Plataforma	SharpDevelop 2.0.1.1710 Microsoft. NET Framework SDK V2.0
Programas	PostgreSQL 8.1

Hardware: Se requiere disponer, con vistas a que el sistema funcione como se tiene concebido, de una computadora con las características que se muestran en la tabla.

Tabla 3: Requisitos de hardware para el adecuado funcionamiento de la aplicación en el cliente.

Componente de Hardware	**Requisitos**
Procesador	2.8 –GHz Pentium IV
Memoria RAM	512 MB o más
Capacidad de almacenamiento en disco duro	80 GB HDD

Tarjeta de Video	VGA o superior

Tabla 4: Requisitos de hardware para el adecuado funcionamiento de la aplicación en el servidor.

Componente de Hardware	**Requisitos**
Procesador	3.2 –GHz Pentium IV
Capacidad de almacenamiento en disco duro	1 Tb HDD

2.7 Modelo del Sistema

2.7.1 Modelo de Casos de Uso del Sistema

Los actores representan los usuarios del sistema y otras aplicaciones que interactúan con él, es decir, representan terceros fuera del sistema que guarda relación con el mismo. Estos suelen corresponderse con trabajadores o actores del negocio. Debido a las características particulares del sistema solamente tienen un actor, el editor. Los casos de usos definidos se relacionan exclusivamente con este actor.

Tabla 5: Actores.

Actores	**Justificación**
Editor	Es el encargado de realizar las ediciones y de actualizar la BD

Para cumplir con los requerimientos se definen los siguientes casos de uso:

Tabla 6: Casos de Uso.

Nombre CU	**Propósito**	**RF**
1. Autenticar usuario	Permitir el acceso al sistema.	R1
2. Añadir efecto	Añadir efectos como: 1.1 Reverse 1.2 Eliminar silencio 1.3 Insertar silencio 1.4 Modificar ganancia 1.5 Silenciar 1.6 Puerta de ruido (noise gate) 1.7 Normalizar 1.8 Aplicar envolvente 1.9 Modular la amplitud 1.10 Modificar la frecuencia de muestreo 1.11 Resemple (remuestreo) 1.12 Transponer	R2
3.Permitir Procesamiento Multipista	Abrir más de una pista a la vez	R3
4. Gestionar Tipo de Audio	Permitir insertar, modificar y eliminar Tipo de audio.	R4
5. Gestionar Carpeta	Permitir insertar, modificar y eliminar carpeta.	R5

6. Gestionar Lista	Permitir insertar, modificar y eliminar lista	R6
7. Gestionar Programa	Permitir insertar, modificar y eliminar programa.	R7
8. Gestionar Elemento	Permitir insertar, modificar y eliminar fichero.	R8

2.7.2 Diagrama de Casos de Uso del sistema

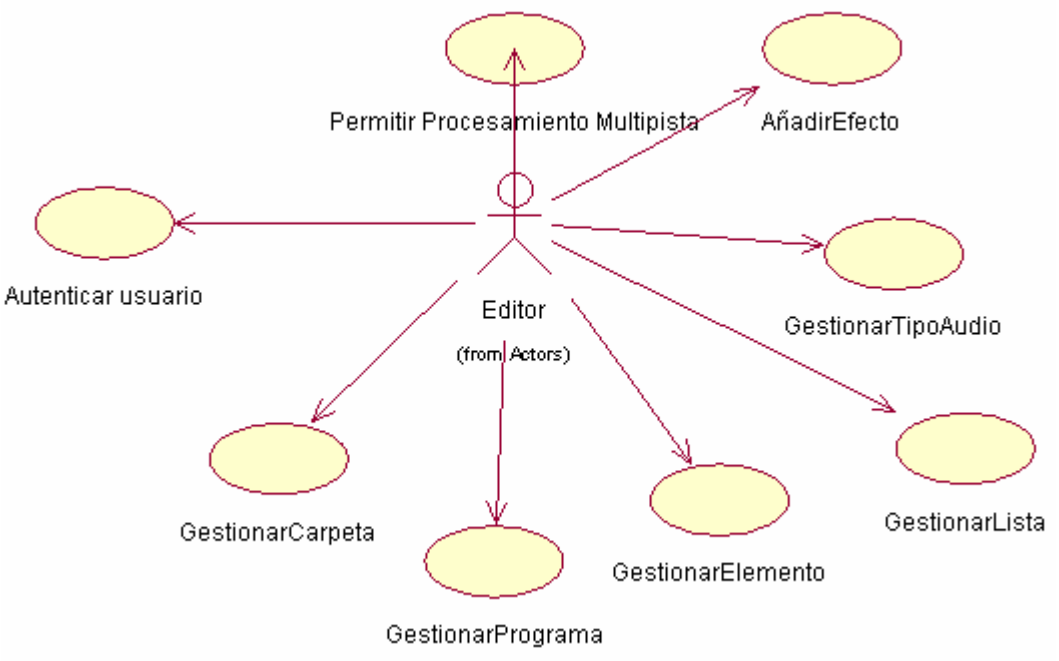


Diagrama 2: Modelo de Casos de Uso del Sistema

2.7.3 Descripción de los CU

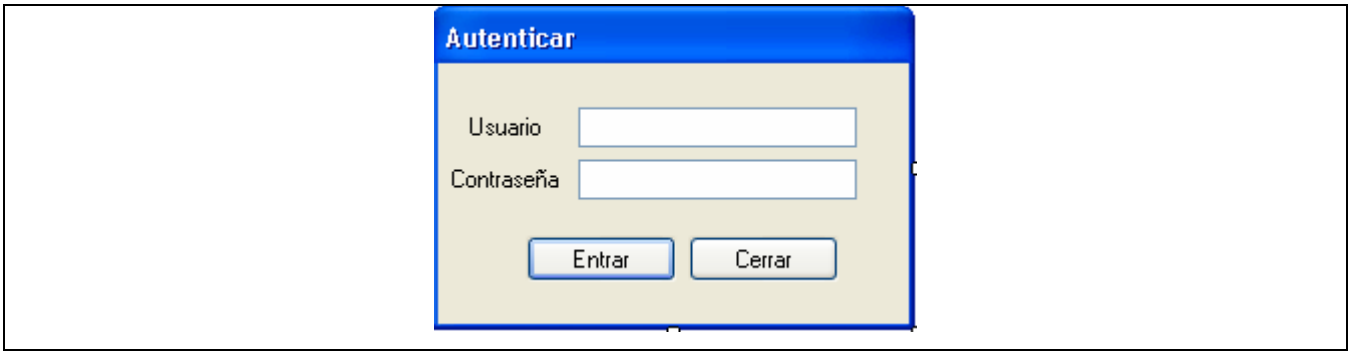
Caso de uso Autenticar usuario

Tabla 7: Caso de uso, Autenticar usuario.

Caso de uso: ¨ Autenticar usuario ¨.	
CU1	Autenticar usuario.
Propósito:	Permitir el acceso al sistema.
Actores:	Editor.
Resumen:	El CU se inicia cuando el editor desea entrar al sistema.
Referencias	R1
Precondiciones:	
Poscondiciones:	Se muestran las funcionalidades del sistema

Curso normal de los eventos.	
Acción del actor:	**Respuesta del sistema**
	1. El sistema muestra el formulario Autenticar Usuario solicitando nombre de usuario y contraseña.
2. El editor introduce los datos solicitados.	
3. El editor presiona el botón Entrar.	3.1 El sistema verifica que los datos sean correctos.

Curso alterno.	
Acción 3	El editor presiona el botón Cerrar.
Acción 3.1	El sistema verifica que los datos, si son incorrectos, muestra un mensaje de error.
Prototipo de Interfaz de usuario	



Caso de uso Añadir efecto

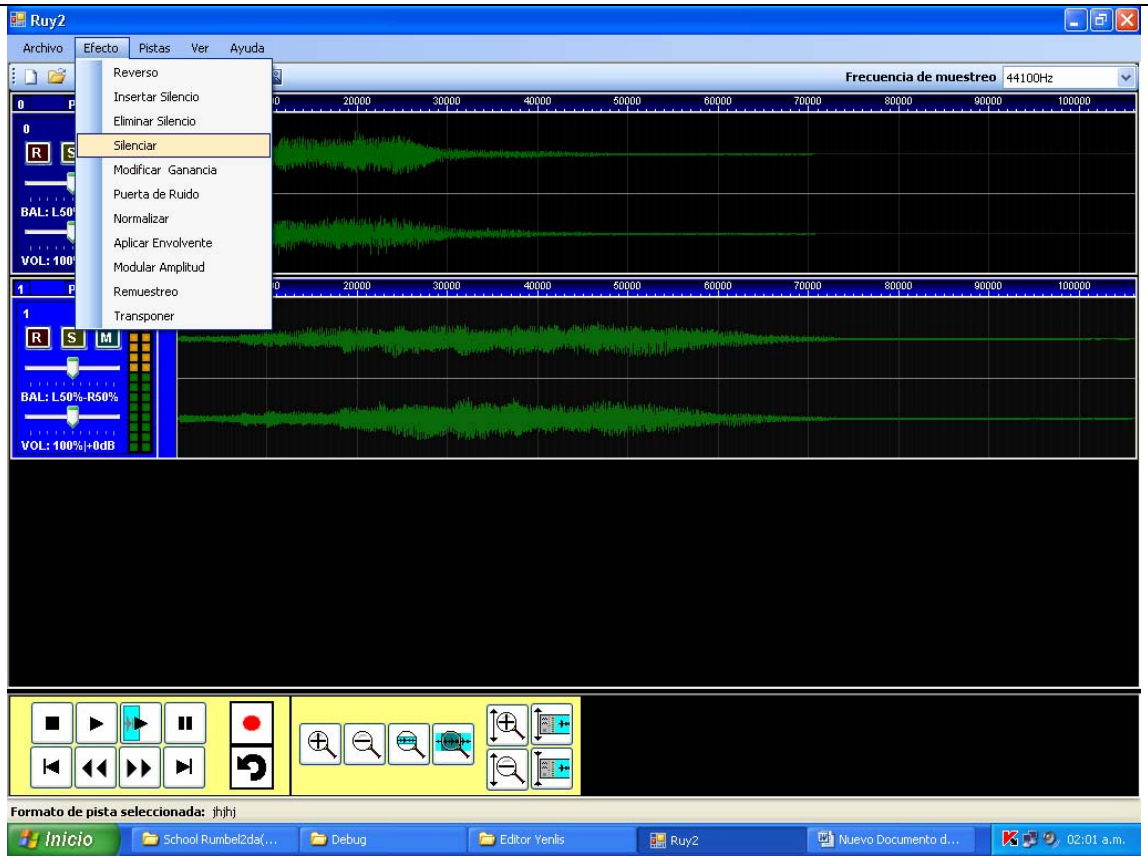
Tabla 8: Caso de uso, Añadir efecto.

Caso de uso: ¨ Añadir efecto ¨.	
CU2	Añadir efecto.
Propósito:	Permite al editor añadir efectos como: 1.1 Reverse 1.2 Eliminar silencio 1.3 Insertar silencio 1.4 Modificar ganancia 1.5 Silenciar 1.6 Puerta de ruido (noise gate)

	1.7 Normalizar 1.8 Aplicar envolvente 1.9 Modular la amplitud 1.10 Modificar la frecuencia de muestreo 1.11 Resemple (remuestreo) 1.12 Transponer
Actores:	Editor.
Resumen:	El CU se inicia cuando el editor selecciona la opción Añadir efecto, para ello el editor debe seleccionar que efecto es el que desea añadir.
Referencias	R2
Precondiciones:	El editor debe haberse autenticado para entrar al sistema y debe haber abierto el fichero al que le desea aplicar el efecto.
Poscondiciones:	

Curso normal de los eventos.	
Acción del actor:	**Respuesta del sistema**
1. El editor selecciona la opción ¨ Efecto.	1.1. Muestra las opciones - Reverse - Eliminar silencio - Insertar silencio - Modificar ganancia - Silenciar - Puerta de ruido (noise gate) - Normalizar - Aplicar envolvente - Modular la amplitud

	- Modificar la frecuencia de muestreo - Resemple (remuestreo) - Transponer
2. El editor selecciona el efecto.	2.1. Aplica el efecto.
Curso alterno.	
Acción 4	Si no llena todos los campos del formulario se retorna a la acción 3.1.
Prototipo de Interfaz de usuario	

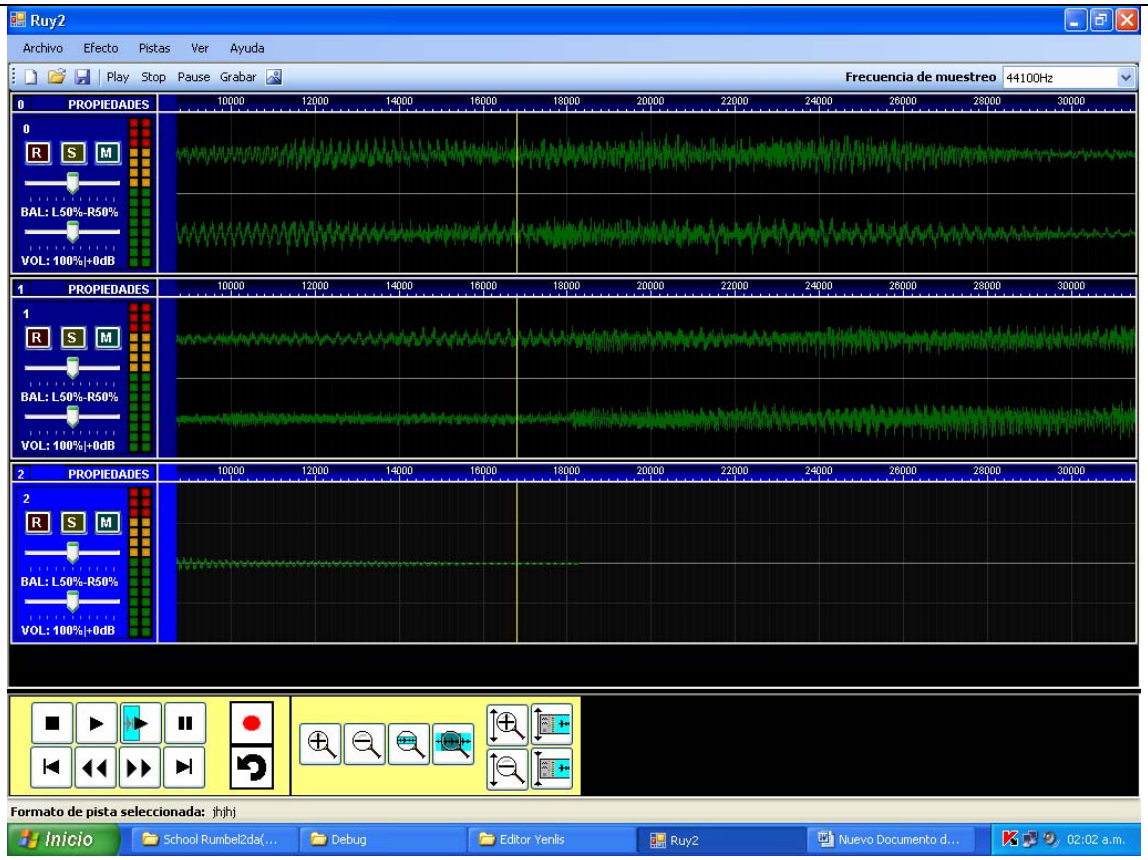


Caso de uso Permitir Procesamiento Multipista

Tabla 9: Caso de uso, Permitir Procesamiento Multipista.

Caso de uso: ¨ Permitir Procesamiento Multipista¨.	
CU3	Permitir Procesamiento Multipista.
Propósito:	Abrir más de una pista a la vez
Actores:	Editor.
Resumen:	El CU se inicia cuando el editor va a abrir un fichero
Referencias	R3
Precondiciones:	El editor debe haberse autenticado para entrar al sistema y debe haber seleccionado un programa.
Poscondiciones:	
Curso normal de los eventos.	
Acción del actor:	**Respuesta del sistema**
1. El editor accede al sistema.	1.1. El sistema le muestra el fichero del elemento que seleccionó.
2. El editor selecciona el fichero y presiona el botón abrir.	2.1 El sistema convierte el fichero a .wav si no es de esta extensión. 2.2. El sistema abre el fichero.
3. El editor selecciona grabar.	3.1. Muestra el formulario, donde le pide que entre el programa, carpeta, autor, interprete, genero, tipo de audio,

	canal, titulo y estado.
4. El editor llena el formulario y presiona el botón aceptar.	4.1 Guarda el nuevo fichero.
5. El editor selecciona el botón cerrar.	5.1 El sistema guarda todos los ficheros que se están abiertos en el estado en que se encuentren, como un fichero temporal .pik.
Curso alterno.	
Acción 4	Si no llena todos los campos del formulario se retorna a la acción 3.1.
Prototipo de Interfaz de usuario	


Ruy2
Archivo
Efecto
Pistas
Ver
Ayuda
Play
Stop
Pause
Grabar
Frecuencia de muestreo
44100Hz
PROPIEDADES
BAL: L50%-R50%
VOL: 100%|+0dB
Formato de pista seleccionada: jhjhj
Inicio
School Rumbel2da(...
Debug
Editor Yenlis
Ruy2
Nuevo Documento d...
02:02 a.m.

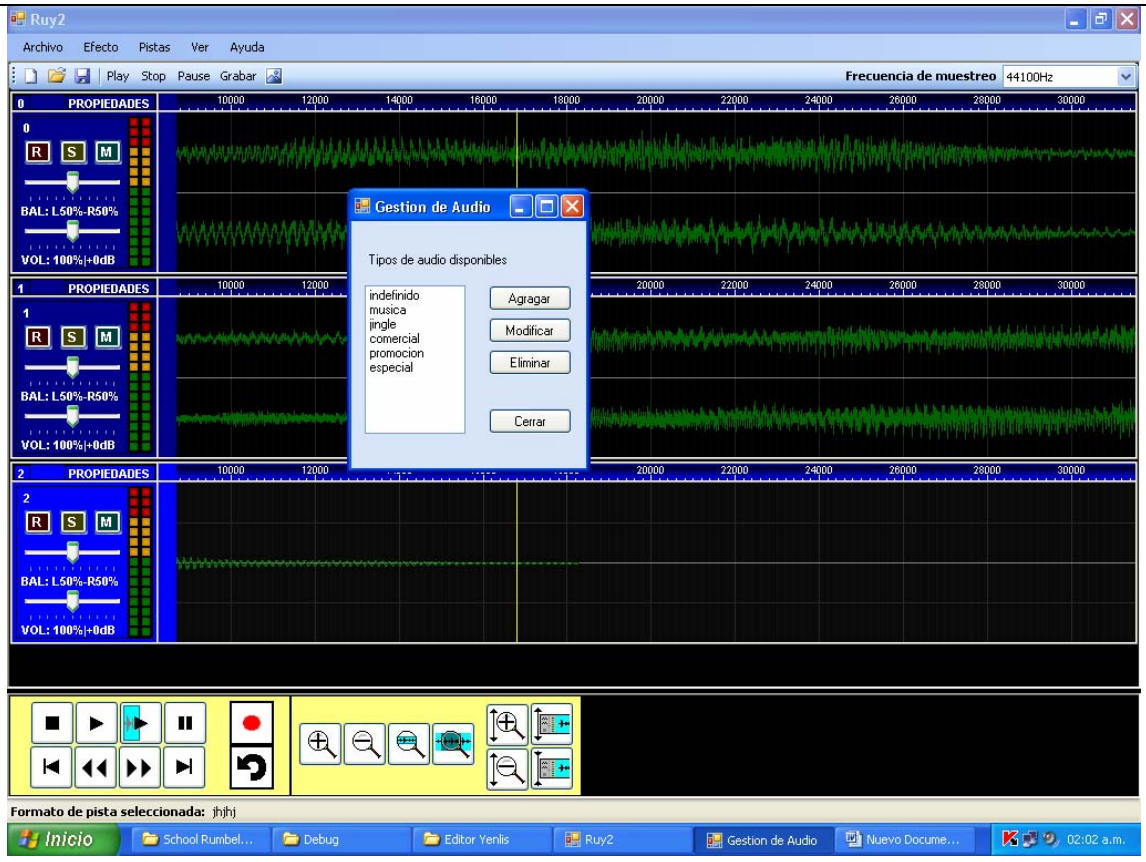
Caso de uso Gestionar TipoAudio

Tabla 10: Caso de uso, Gestionar TipoAudio

Caso de uso: ¨ Gestionar TipoAudio.	
CU4	Gestionar TipoAudio.
Propósito:	Permitir insertar, modificar y eliminar un tipo de audio.
Actores:	Editor.
Resumen:	El CU se inicia cuando el editor desea insertar, modificar o eliminar un tipo de audio.
Referencias	R4
Precondiciones:	El editor debe haberse autenticado para entrar al sistema.
Poscondiciones:	
Curso normal de los eventos.	
Acción del actor:	**Respuesta del sistema**
1. El editor accede al sistema para Gestionar TipoAudio.	1.1 El sistema le muestra los Tipos de audio disponibles y las operaciones que pueden realizarse a) Agregar. b) Eliminar. c) Modificar.
Escenario 1: ¨ Agregar ¨**.**	
Acción del actor:	**Respuesta del sistema:**

2. El editor selecciona la opción Agregar.	2.1. Le muestra el formulario para que entre el nombre del nuevo tipo de audio.
3. El editor llena el formulario que se le muestra y presiona el botón Aceptar.	3.1. Inserta en la BD el nuevo formato.
Curso alterno.	
Acción 3	Si no llena el formulario, o si el tipo de audio que desea insertar ya existe o si presiona el botón cerrar se retorna a la acción 2.1.
Escenario 2: ¨ Eliminar ¨**.**	
Acción del actor	**Respuesta del sistema**
	2. Le muestra los un tipo de audio para que seleccione cual Tipo de Audio desea eliminar.
3. El editor selecciona el TipoAudio que desea eliminar y presiona el botón Eliminar.	3.1. Elimina el Tipo Audio.
Curso alterno.	
Acción 3	Si presiona el botón cerrar se retorna a la acción 2.1.
Escenario 2: ¨ Modificar ¨**.**	
Acción del actor	**Respuesta del sistema**
2. El editor selecciona la opción Modificar.	2.1. Le muestra los Tipo de Audio para que seleccione cual desea modificar y para que entre el nuevo nombre del Tipo de Audio.
3. El editor selecciona el Tipo de Audio que desea modificar y el nuevo nombre del Tipo de Audio y presiona el botón Aceptar.	3.1. Modifica el Tipo de Audio.

Curso alterno.	
Acción 3	Si presiona el botón cerrar o si no entra el nuevo nombre se retorna a la acción 2.1.
Prototipo de Interfaz de usuario	



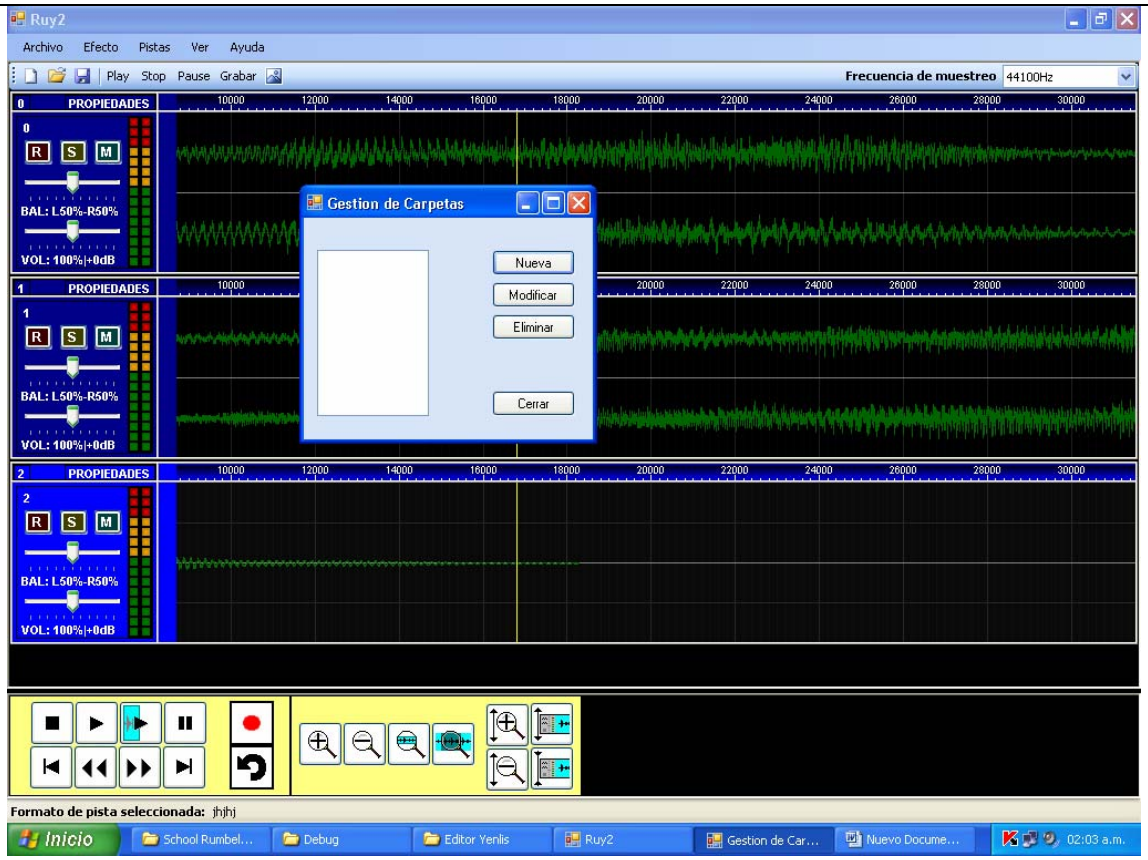
Caso de uso Gestionar Carpeta

Tabla 11: Caso de uso, Gestionar Carpeta.

Caso de uso: ¨ Gestionar Carpeta ¨.	
CU5	Gestionar Carpeta.
Propósito:	Permitir insertar, modificar y eliminar carpeta.
Actores:	Editor.
Resumen:	El CU se inicia cuando el editor desea insertar, modificar o eliminar una carpeta.
Referencias	R5
Precondiciones:	El editor debe haberse autenticado para entrar al sistema y debe haber seleccionado un programa.
Poscondiciones:	
Curso normal de los eventos.	
Acción del actor:	**Respuesta del sistema**
1. El editor accede al sistema para Gestionar carpetas.	1.1 El sistema muestra las carpetas del programa seleccionado y las operaciones que pueden realizarse d) Nueva. e) Eliminar. f) Modificar.
Escenario 1: ¨ Nueva ¨**.**	
Acción del actor:	**Respuesta del sistema:**

2. El editor selecciona la opción Nuevo.	2.1. Le muestra el formulario que debe llenar para insertarlo en la BD, en el mismo debe poner el nombre de la carpeta.
3. El editor llena el formulario que se le muestra y presiona el botón Aceptar.	3.1. Inserta en la BD la nueva carpeta.
Curso alterno.	
Acción 3	Si no llena el formulario completo, o si la carpeta que desea insertar ya existe o si presiona el botón Cerrar se retorna a la acción 2.1.
Escenario 2: ¨ Eliminar ¨**.**	
Acción del actor	**Respuesta del sistema**
	2. Le muestra las carpetas que contiene el programa seleccionado, para que seleccione que carpeta desea eliminar.
3. El editor selecciona la carpeta que desea eliminar y presiona el botón eliminar.	3.1. Elimina la carpeta.
Curso alterno.	
Acción 3	Si presiona el botón Cerrar se retorna a la acción 2.
Escenario 3: ¨ Modificar ¨**.**	
Acción del actor	**Respuesta del sistema**
2. El editor selecciona la opción Modificar.	2.1. Le muestra las carpetas que contiene el programa seleccionado para que seleccione cual carpeta desea modificar y le pide que entre el nuevo nombre.
3. El editor selecciona la carpeta que desea modificar, entra el nombre y presiona el botón	3.1. Modifica la carpeta.

aceptar.	
Curso alterno.	
Acción 3	Si presiona el botón Cerrar o si no entra el nuevo nombre se retorna a la acción 2.1.
Prototipo de Interfaz de usuario	



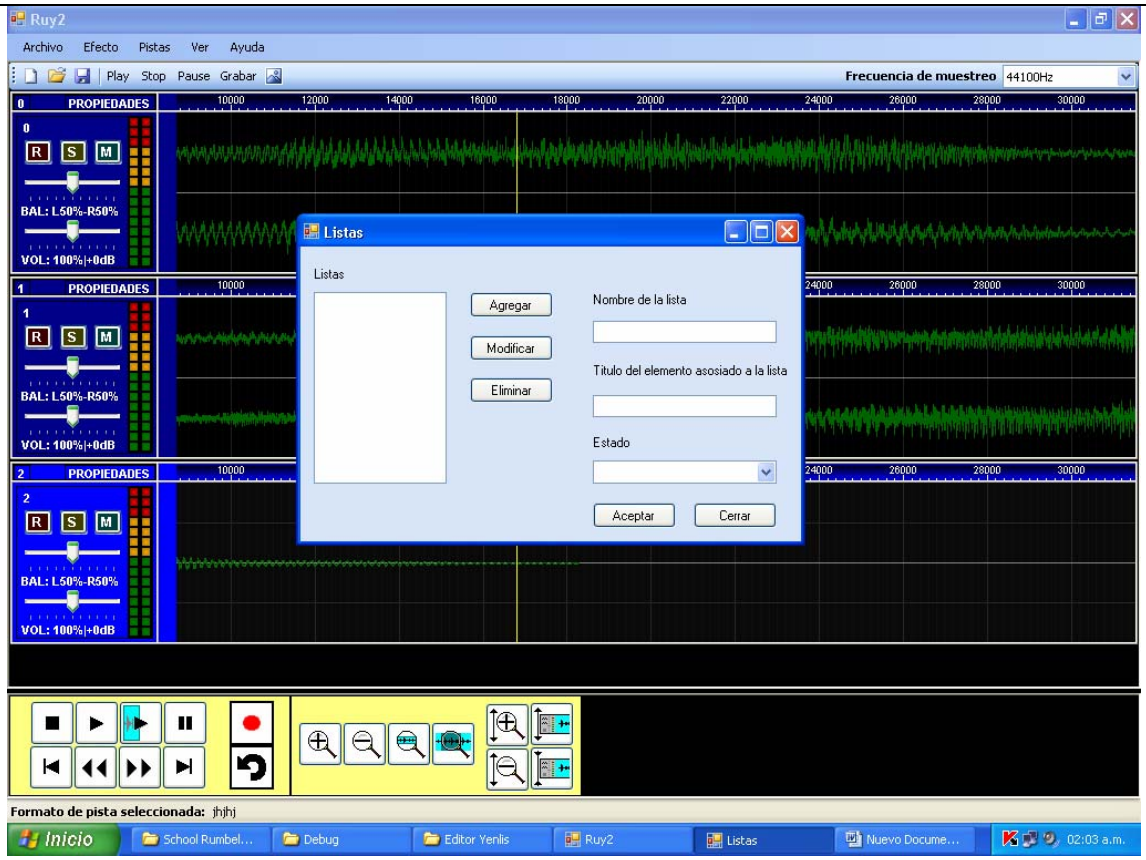
Caso de uso Gestionar Lista

Tabla 12: Caso de uso, Gestionar Lista.

Caso de uso: ¨ Gestionar Lista ¨.	
CU6	Gestionar Lista.
Propósito:	Permitir insertar, modificar y eliminar lista.
Actores:	Editor.
Resumen:	El CU se inicia cuando el editor desea insertar, modificar o eliminar una lista.
Referencias	R6
Precondiciones:	El editor debe haberse autenticado para entrar al sistema.
Poscondiciones:	
Curso normal de los eventos.	
Acción del actor:	**Respuesta del sistema**
1. El editor accede al sistema para Gestionar lista.	1.1 El sistema le permite que seleccione la lista con la que desea trabajar, las carpetas de la BD, con sus elementos correspondientes y las operaciones que pueden realizarse g) Agregar. h) Eliminar. i) Modificar.
Escenario 1: ¨ Agregar ¨**.**	
Acción del actor:	**Respuesta del sistema:**
2. El editor selecciona la opción	2.1. Le muestra el formulario que debe llenar para insertarlo en

Agregar.	la BD, en el mismo debe poner el nombre de la lista y del nuevo elemento debe poner el título y el estado.
3. El editor llena el formulario que se le muestra y presiona el botón Aceptar.	3.1. Inserta en la BD la nueva lista.
Curso alterno.	
Acción 3	Si no llena el formulario, o si la lista que desea agregar ya existe, o si presiona el botón Cerrar se retorna a la acción 2.1.
Escenario 2: ¨ Eliminar lista ¨**.**	
Acción del actor	**Respuesta del sistema**
	2. Le muestra las listas que existen, para que seleccione que lista desea eliminar.
3. El editor selecciona la lista que desea eliminar y presiona el botón Eliminar.	3.1. Elimina la lista.
Curso alterno.	
Acción 3	Si presiona el botón cerrar se retorna a la acción 2.1.
Escenario 3: ¨ Modificar ¨**.**	
Acción del actor	**Respuesta del sistema**
2. El editor selecciona la opción Modificar.	2.1. Le muestra las listas que existen para que seleccione cual lista desea modificar y le muestra el elemento que estas asociado a ella por si quiere modificarle los datos del elemento.
3. El editor selecciona la lista que desea modificar, si desea modificarle los datos del elemento asociado o puede cambiarle el	3.1. Modifica la lista.

nombre a la lista y presionar el botón Aceptar.	
Curso alterno.	
Acción 3	Si presiona el botón Cerrar se retorna a la acción 2.1.
Prototipo de Interfaz de usuario	



Caso de uso Gestionar Programa

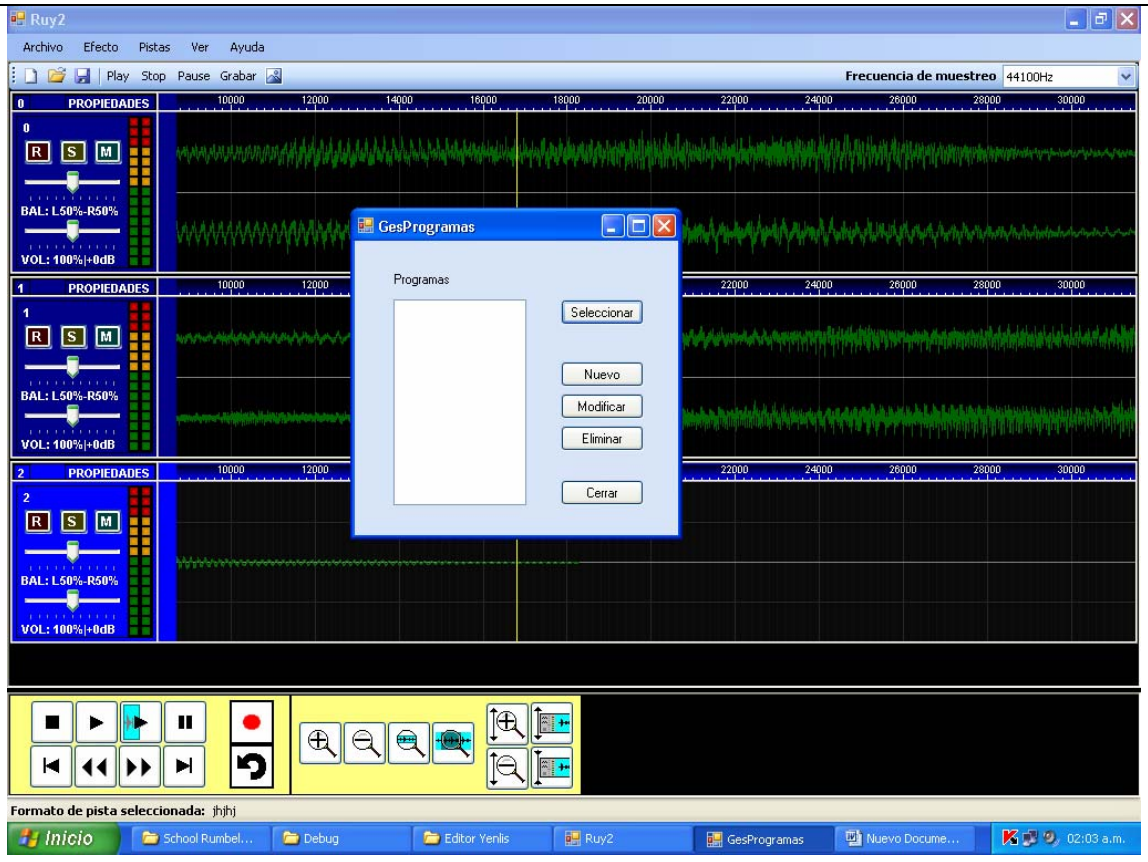
Tabla 13: Caso de uso, Gestionar Programa

.

Caso de uso: ¨ Gestionar Programa¨.	
CU7	Gestionar Programa
Propósito:	Permitir insertar, modificar y eliminar un programa.
Actores:	Editor.
Resumen:	El CU se inicia cuando el editor desea insertar, modificar o eliminar un programa.
Referencias	R7
Precondiciones:	El editor debe haberse autenticado para entrar al sistema.
Poscondiciones:	
Curso normal de los eventos.	
Acción del actor:	**Respuesta del sistema**
1. El editor accede al sistema para Gestionar programa.	1.1 El sistema le muestra los programas que existen y las operaciones que pueden realizarse j) Nuevo. k) Eliminar. l) Modificar. m) Seleccionar
Escenario 1: ¨ Nuevo ¨**.**	

Acción del actor:	**Respuesta del sistema:**
2. El editor selecciona la opción Nuevo.	2.1. Le muestra el formulario que debe llenar para insertarlo en la BD, en el mismo debe poner el nombre del programa.
3. El editor llena el formulario que se le muestra y presiona el botón Aceptar.	3.1. Inserta en la BD el nuevo programa.
Curso alterno.	
Acción 3	Si no llena el formulario, o si el programa que desea insertar ya existe, o si presiona el botón Cerrar se retorna a la acción 2.1.
Escenario 2: ¨ Eliminar ¨**.**	
Acción del actor	**Respuesta del sistema**
	2. Le muestra los programas que existen para que seleccione que programa desea eliminar.
3. El editor selecciona el programa que desea eliminar y presiona el botón Eliminar.	3.1. Elimina el programa.
Curso alterno.	
Acción 3	Si presiona el botón Cerrar se retorna a la acción 2.1.
Escenario 3: ¨ Modificar ¨**.**	
Acción del actor	**Respuesta del sistema**
2. El editor selecciona la opción Modificar.	2.1. Le muestra los programas Que existen para que seleccione cual programa desea modificar y le pide que entre el nuevo nombre.
3. El editor selecciona el programa que desea modificar, entra el nombre y presiona el botón	3.1. Modifica el programa.

Aceptar.	
Curso alterno.	
Acción 3	Si presiona el botón Cerrar o si no entra el nuevo nombre se retorna a la acción 2.1.
Escenario 4: ¨ Seleccionar ¨**.**	
2. El editor selecciona la opción Seleccionar.	2.1. Guarda en memoria el id del programa que hemos seleccionado.
Prototipo de Interfaz de usuario	



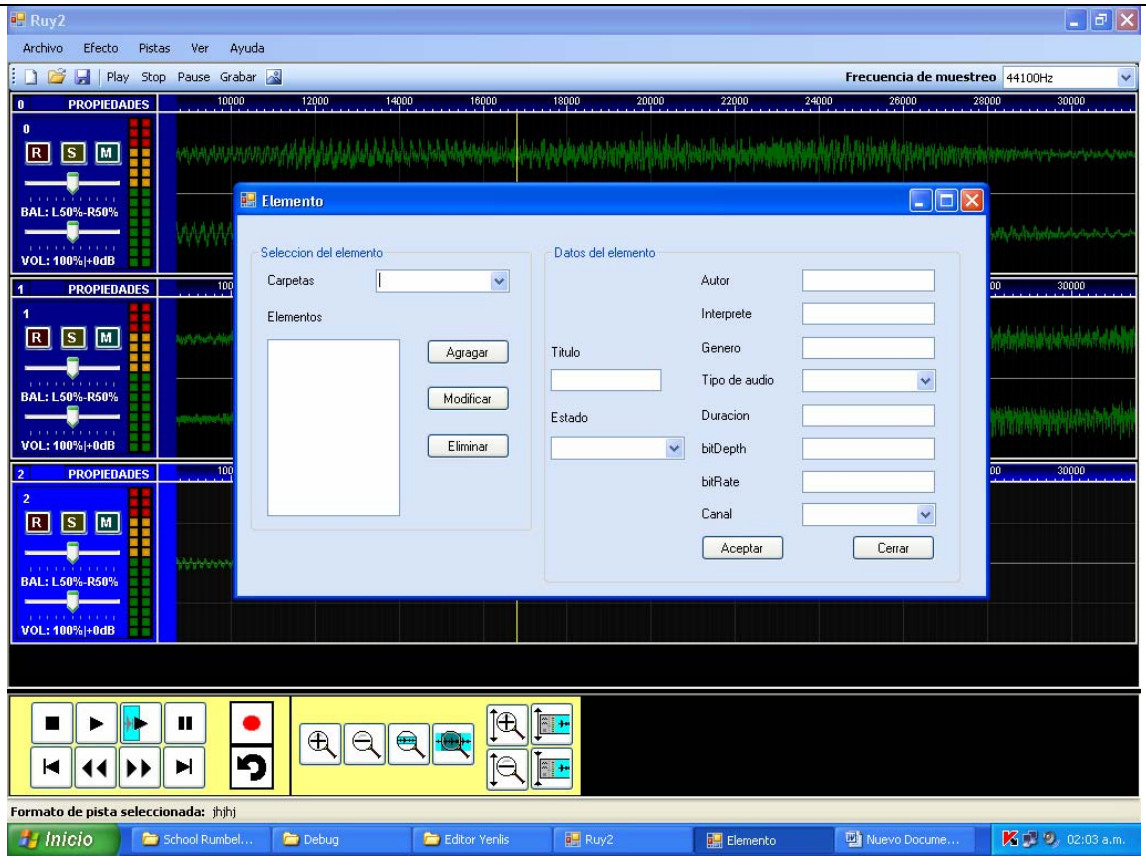
Caso de uso Gestionar Elemento

Tabla 14: Caso de uso, Gestionar Elemento.

Caso de uso: ¨ Gestionar elemento ¨.	
CU8	Gestionar Elemento.
Propósito:	Permitir insertar, modificar y eliminar el elemento.
Actores:	Editor.
Resumen:	El CU se inicia cuando el editor desea actualizar la BD, ya sea para eliminar, modificar o para insertar un fichero.
Referencias	R8
Precondiciones:	El editor debe haberse autenticado para entrar al sistema, debe haber seleccionado un programa y una carpeta.
Poscondiciones:	
Curso normal de los eventos.	
Acción del actor:	**Respuesta del sistema**
1. El editor accede al sistema para Gestionar Elemento.	1.1 El sistema le muestra los elementos de la carpeta seleccionada y las opciones referentes a las operaciones que pueden realizarse n) Agregar. o) Modificar. p) Eliminar.
Escenario 1: ¨ Agregar ¨**.**	

Acción del actor:	**Respuesta del sistema:**
2. El editor selecciona la opción Agregar.	2.1. Le muestra el formulario, donde le pide que entre el titulo y el estado del elemento además los datos del fichero asociado que son título, autor, intérprete, género, formato, canal.
3. El editor llena el formulario que se le muestra y presiona el botón aceptar.	3.1. Inserta en la BD el elemento.
Curso alterno.	
Acción 3	Si no entra el título y el estado del elemento o el titulo del fichero o si presiona el botón Cerrar se retorna a la acción 2.1.
Escenario 2: ¨ Eliminar ¨**.**	
Acción del actor	**Respuesta del sistema**
	2. Le muestra las elementos de la carpeta seleccionada para que seleccione cual elemento desea eliminar.
3. El editor selecciona el elemento y presiona el botón Eliminar.	3.1. Elimina el elemento.
Curso alterno.	
Acción 3	Si presiona el botón cancelar se retorna a la acción 2.
Escenario 2: ¨ Modificar ¨**.**	
Acción del actor	**Respuesta del sistema**
2. El editor selecciona la opción Modificar.	2.1. Le muestra los elemento de la carpeta para que seleccione que elemento desea modificar, para ello debe llenar el formulario, donde le pide que entre el titulo y estado, además puede cambiar los datos del fichero para ello debe entrar los nuevos valores como: autor, interprete, género, formato, canal.

3. El editor selecciona el elemento que desea modificar, llena el formulario y presiona el botón Aceptar.	3.1. Modifica el elemento.
Curso alterno.	
Acción 3	Si presiona el botón Cerrar se retorna a la acción 2.1.
Prototipo de Interfaz de usuario	



2.9 Conclusiones

El capítulo muestra las peculiaridades y características que tiene que presentar el software para cumplir los requerimientos especificados por el cliente. Para describir el contexto en que se desenvuelve se realiza el modelo del dominio. Se definieron los requisitos funcionales y no funcionales que debe cumplir, se elaboró el diagrama de casos de uso del sistema, con su descripción.

Capítulo 3. Análisis y diseño del sistema

3.1 Introducción

En este capítulo se define como se va a desarrollar la aplicación, a través de los artefactos de análisis y diseño, aplicando los principios de diseño.

El diagrama de clases contiene la colección de los elementos estáticos y dinámicos del sistema, como clases, tipos y sus relaciones, conectados unos a otros y a sus contenidos.

3.2 Análisis

El análisis consiste en transformar los requisitos funcionales en un diseño de clases en el cual se vean las relaciones e interacción que existe entre ellas, teniendo en cuenta en este proceso una arquitectura robusta que permita adaptar el sistema al entorno de implementación que se esta desarrollando. Además en este flujo se obtiene una visión del sistema que se preocupa de ver que hace, de tal forma que se preocupa solo por los requisitos funcionales.

3.2.1 Diagrama de Clases del Análisis.

DIAGRAMA DE CLASES DE ANÁLISIS CU_ Autenticar usuario

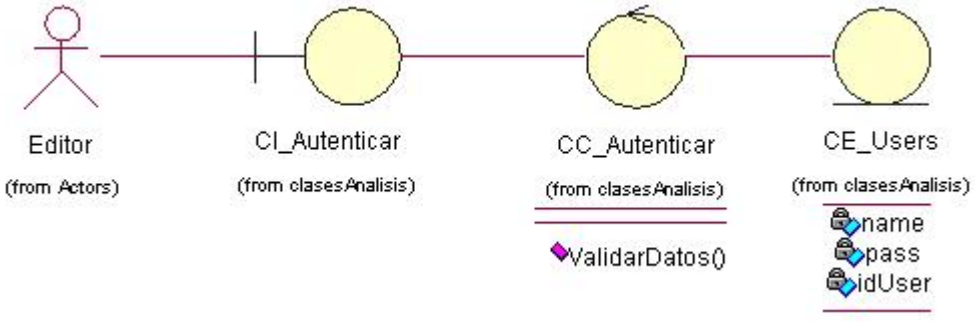


DIAGRAMA DE CLASES DE ANÁLISIS CU_GestionarCarpeta

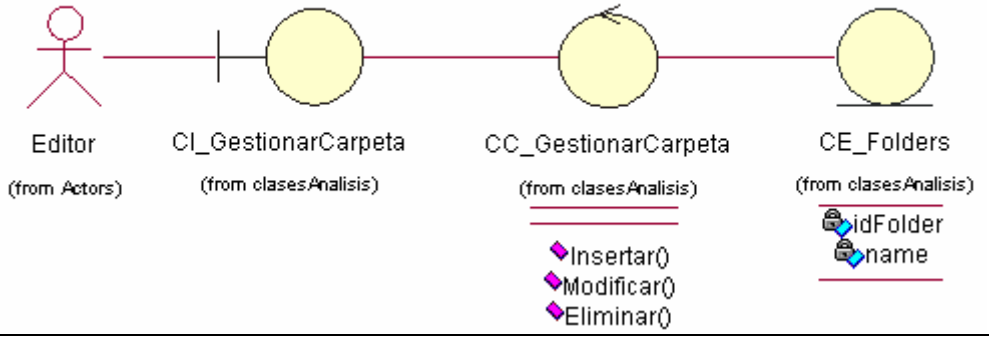


DIAGRAMA DE CLASES DE ANÁLISIS CU_AñadirEfecto

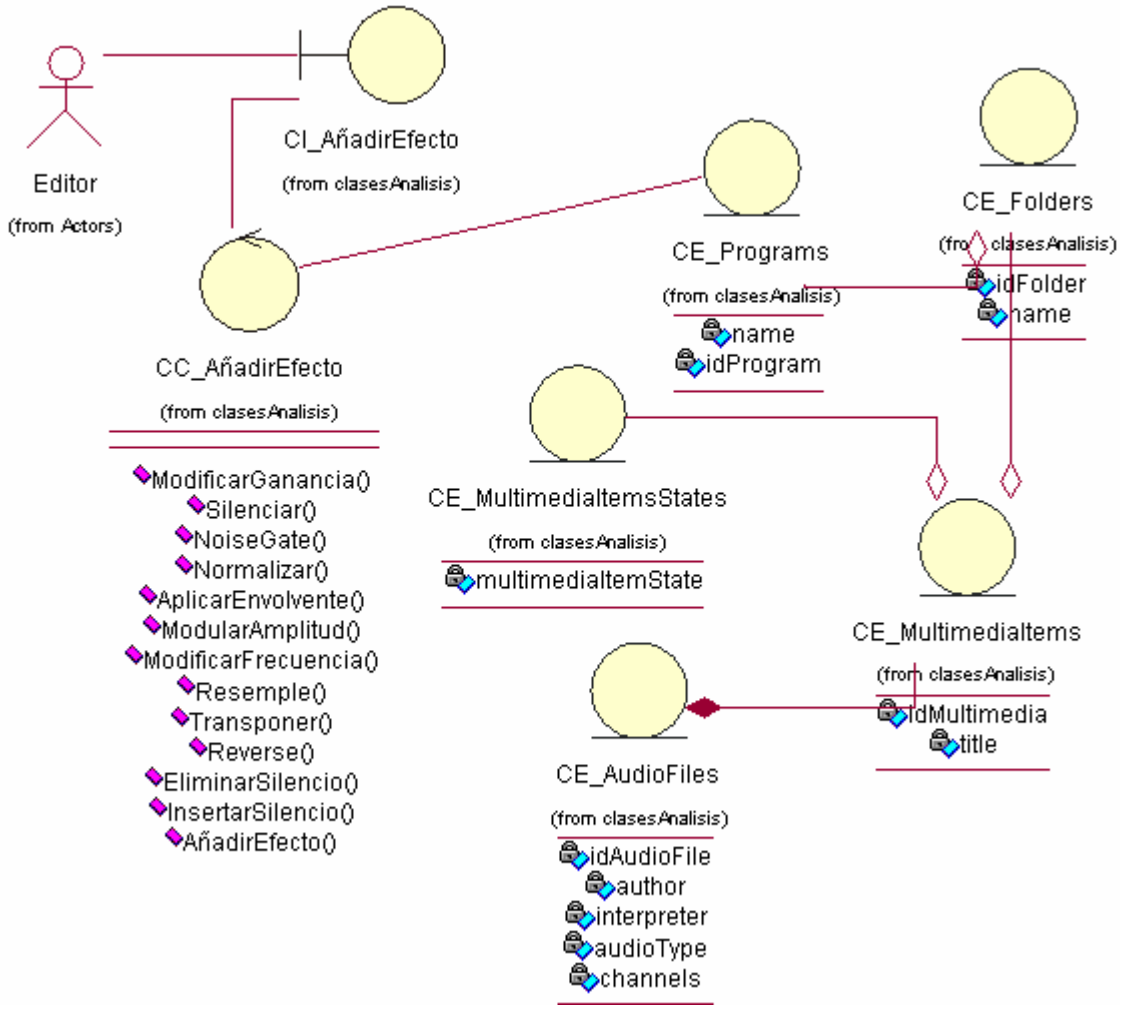


DIAGRAMA DE CLASES DE ANÁLISIS CU_GestionarElemento

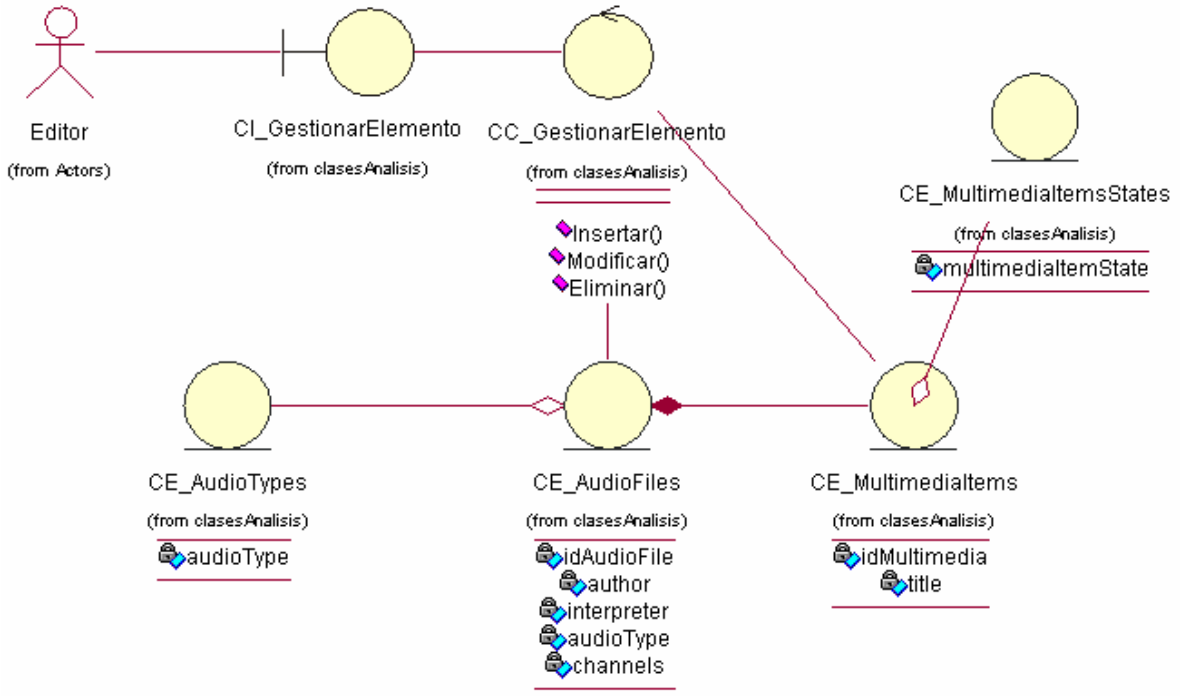


DIAGRAMA DE CLASES DE ANÁLISIS CU_GestionarTipoAudio

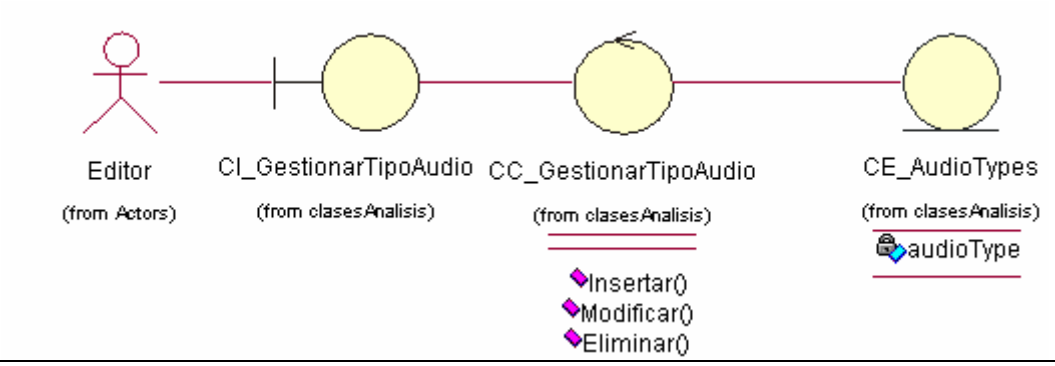


DIAGRAMA DE CLASES DE ANÁLISIS CU_GestionarLista

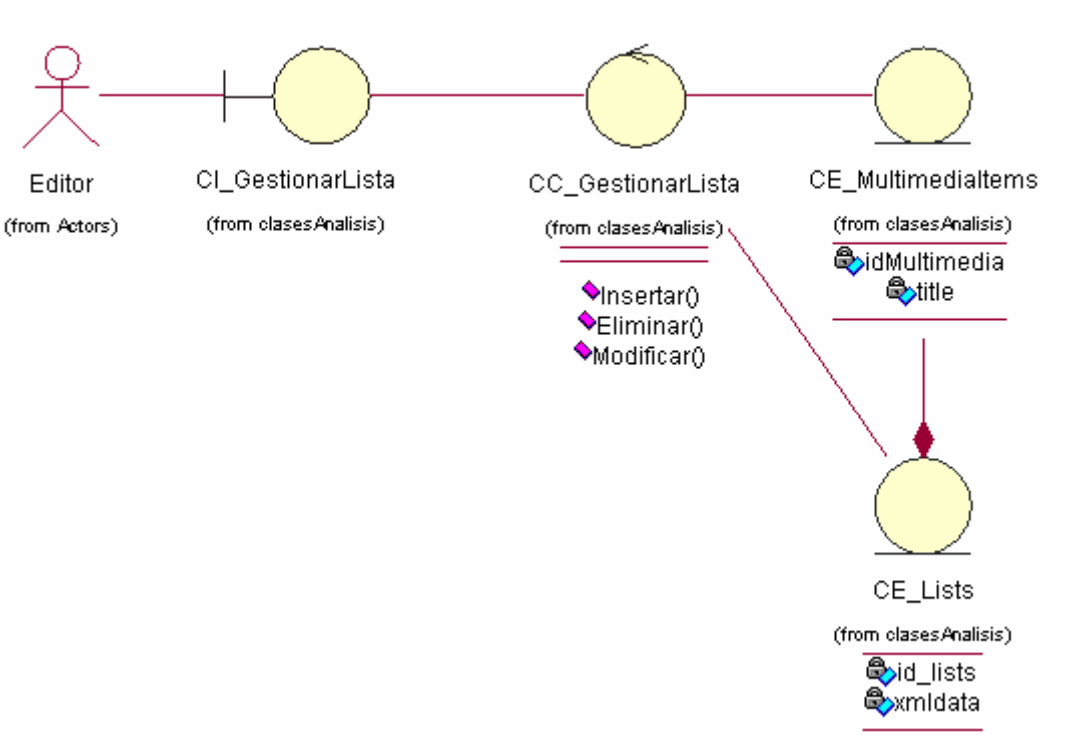


DIAGRAMA DE CLASES DE ANÁLISIS CU_GestionarPrograma

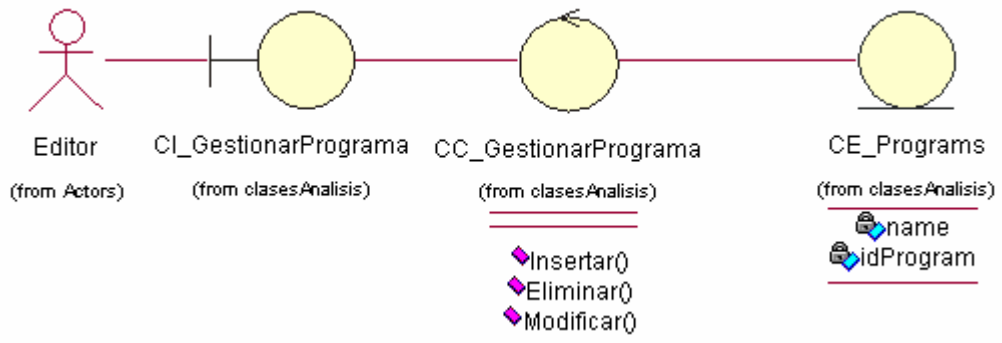
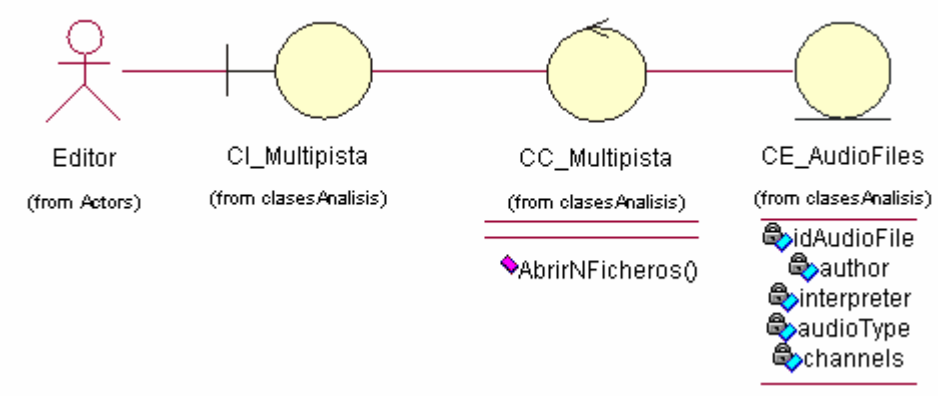


DIAGRAMA DE CLASES DE ANÁLISIS CU_Permitir Procesamiento Multipista



3.3 Diseño

3.3.1 Diagramas de Interacción [Ver Anexo 1]

Los diagramas de interacción no son más que una descripción del modo en el que cada operación detectada en los diagramas de secuencia lleva a cabo sus responsabilidades y modifica el estado del sistema. En UML los diagramas de interacción pueden representarse a través de los Diagramas de Colaboración y/o de los Diagramas de Secuencia.

El tipo de diagrama seleccionado para construir los diagramas de interacción fue el de Secuencia, debido a que muestra cómo los objetos se comunican unos con otros en una secuencia de tiempo, qué sucede en cada momento, y para ello contienen objetos con sus ciclos de vida y los mensajes que se envían entre ellos ordenados secuencialmente. [16]

El diagrama de secuencias es el núcleo de un modelo dinámico, y muestra todos los cursos alternos que pueden tomar los casos de uso. Los diagramas de secuencias se componen de 4 elementos que son: el curso de acción, los objetos, los mensajes y los métodos.

3.3.2 Diagrama de Clases del diseño

DIAGRAMA DE CLASES CU_Autenticar usuario

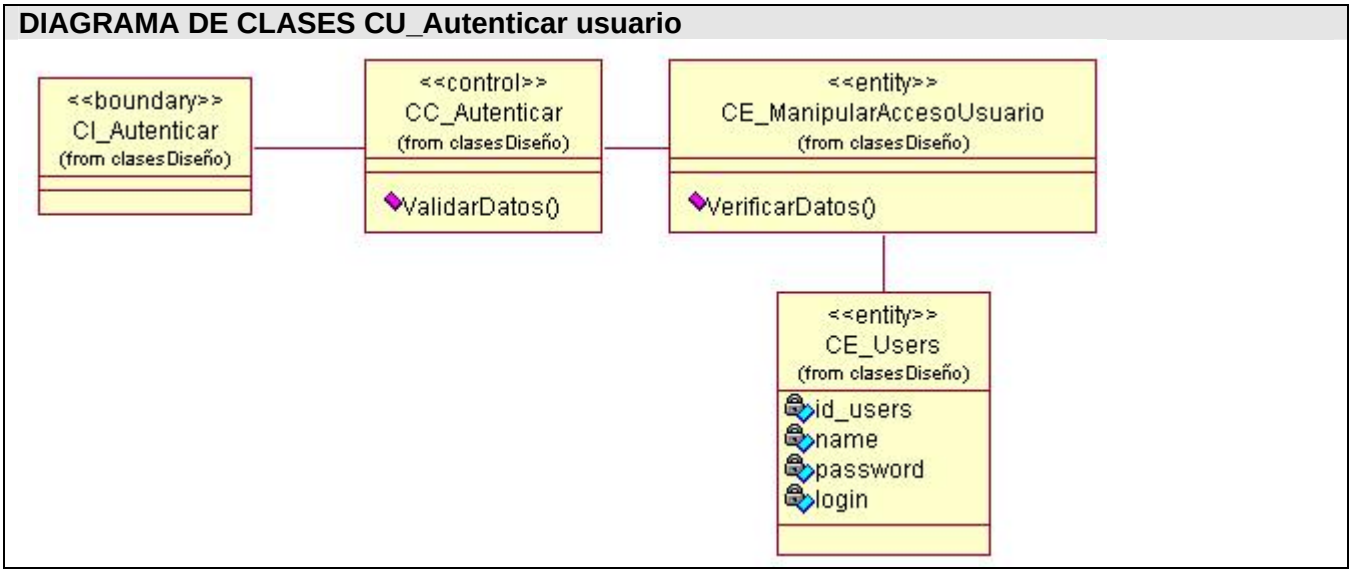


DIAGRAMA DE CLASES CU_AñadirEfecto

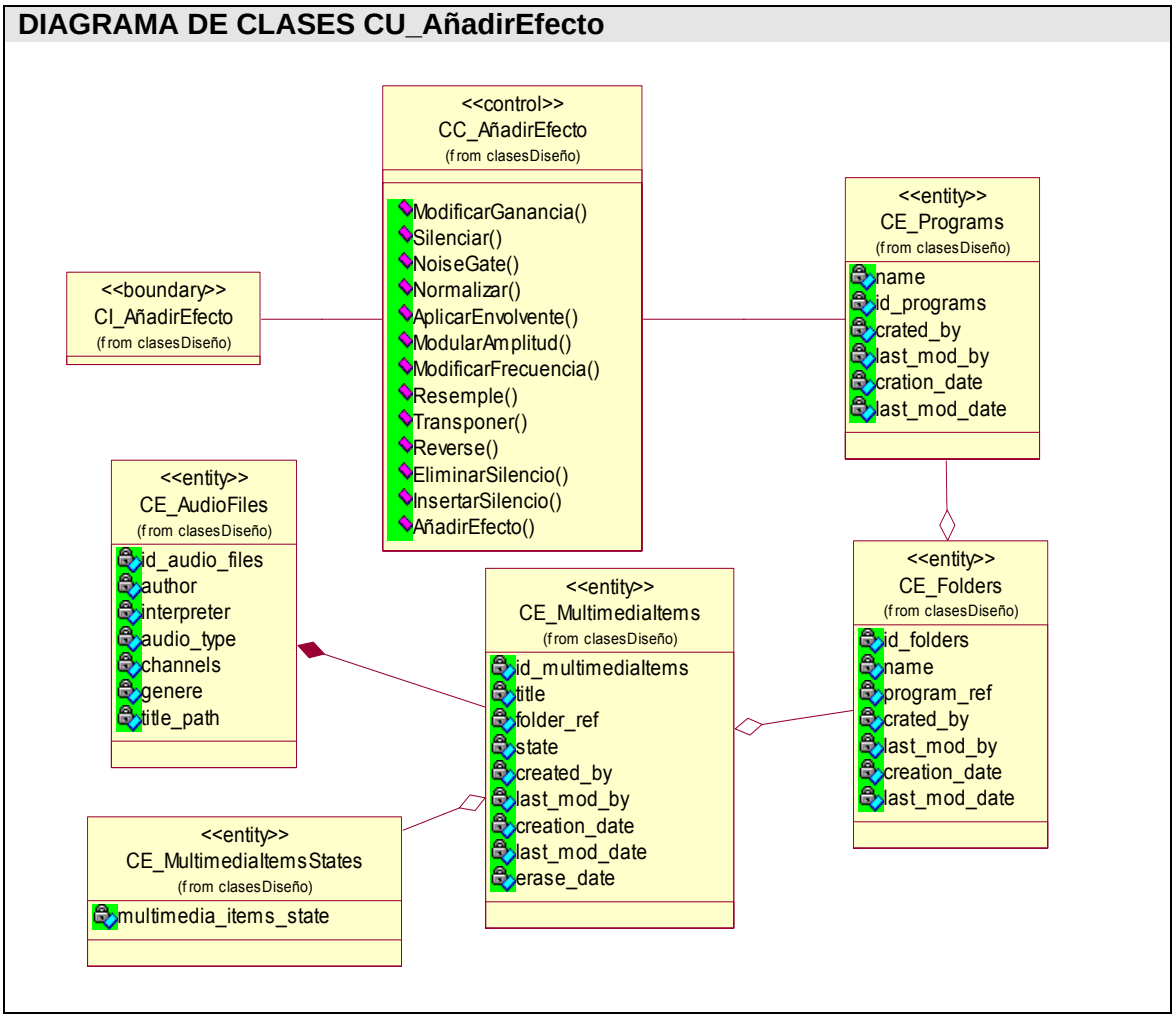


DIAGRAMA DE CLASES CU_GestionarCarpeta

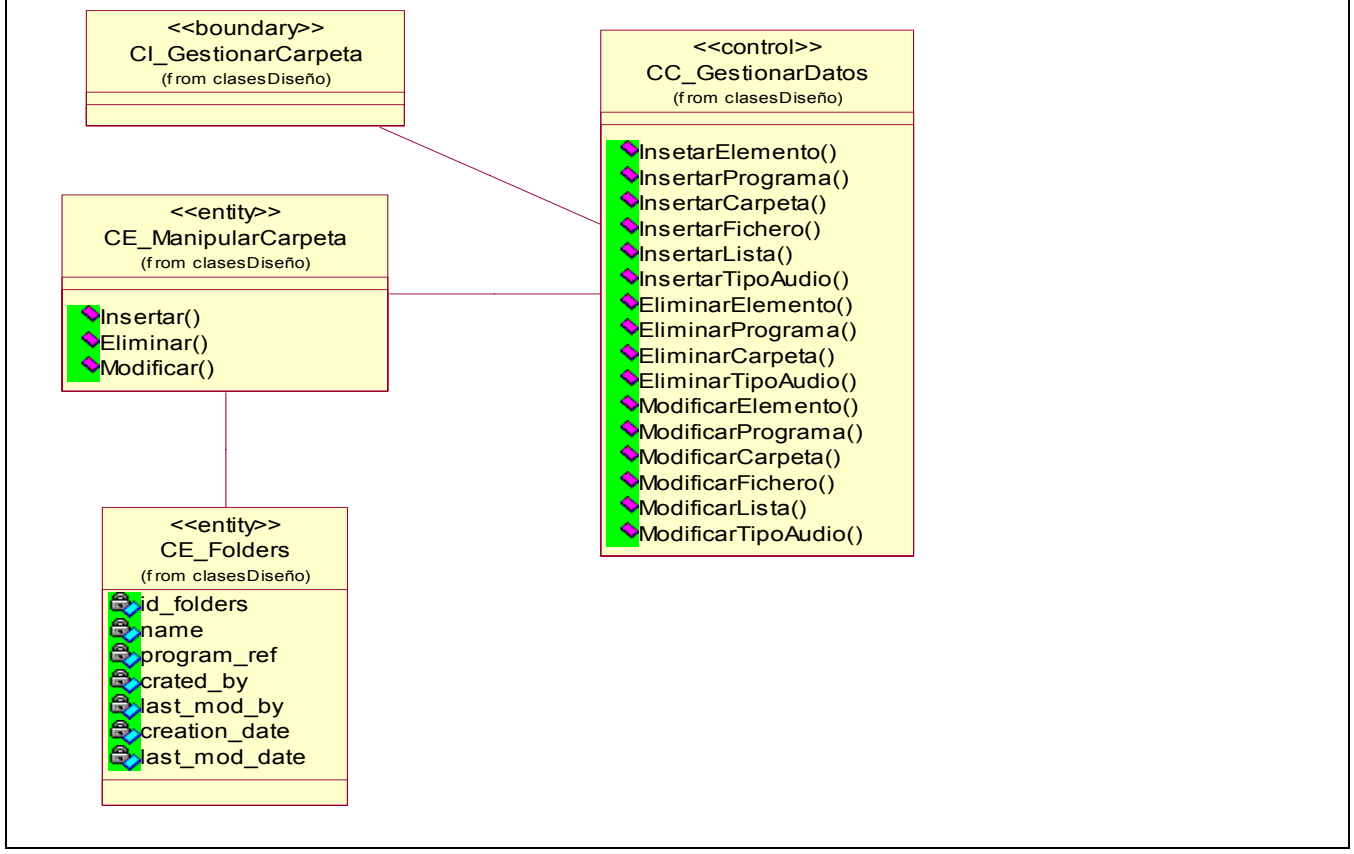


DIAGRAMA DE CLASES CU_GestionarElemento

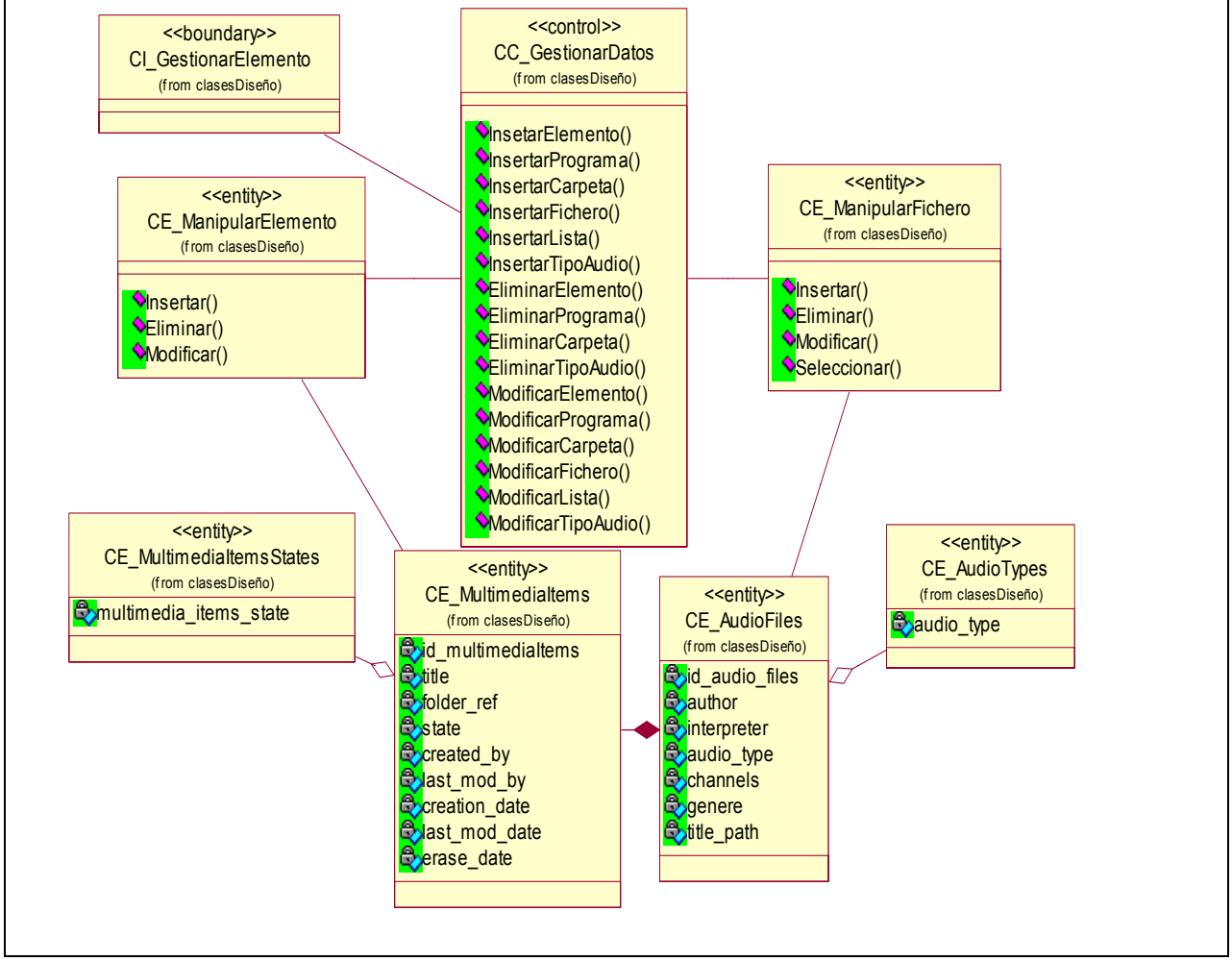


DIAGRAMA DE CLASES CU_GestionarTipoAudio

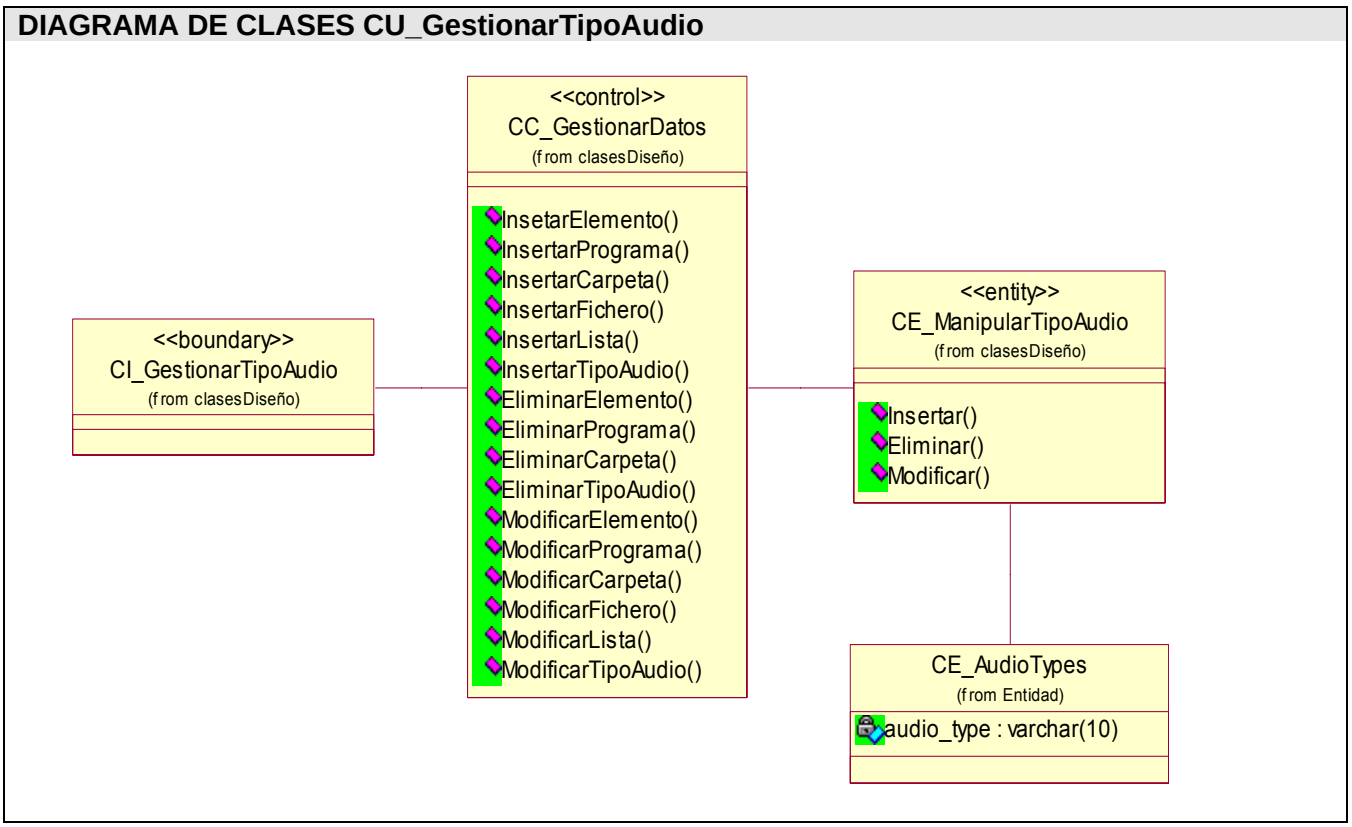


DIAGRAMA DE CLASES CU_GestionarLista

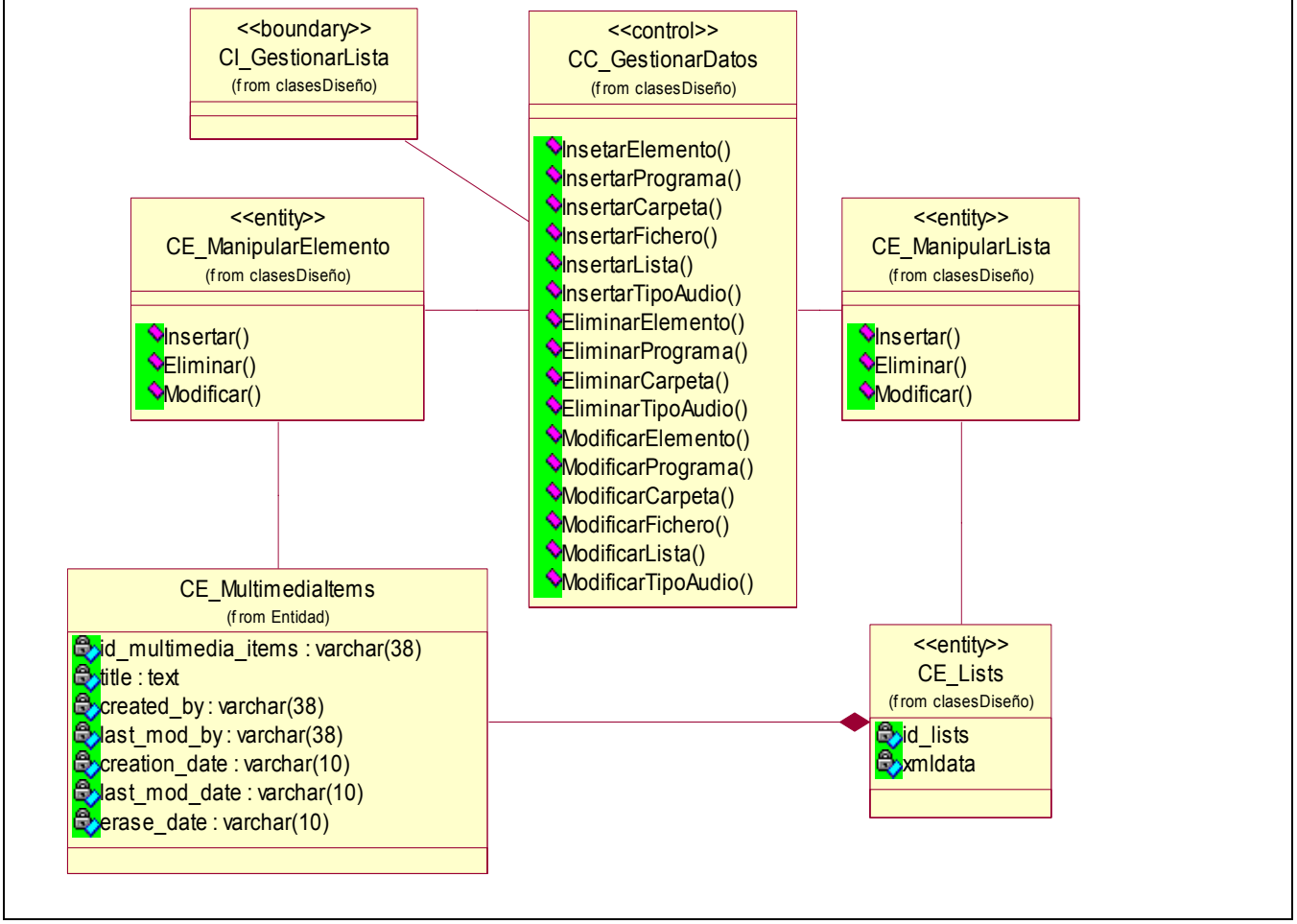


DIAGRAMA DE CLASES CU_GestionarPrograma

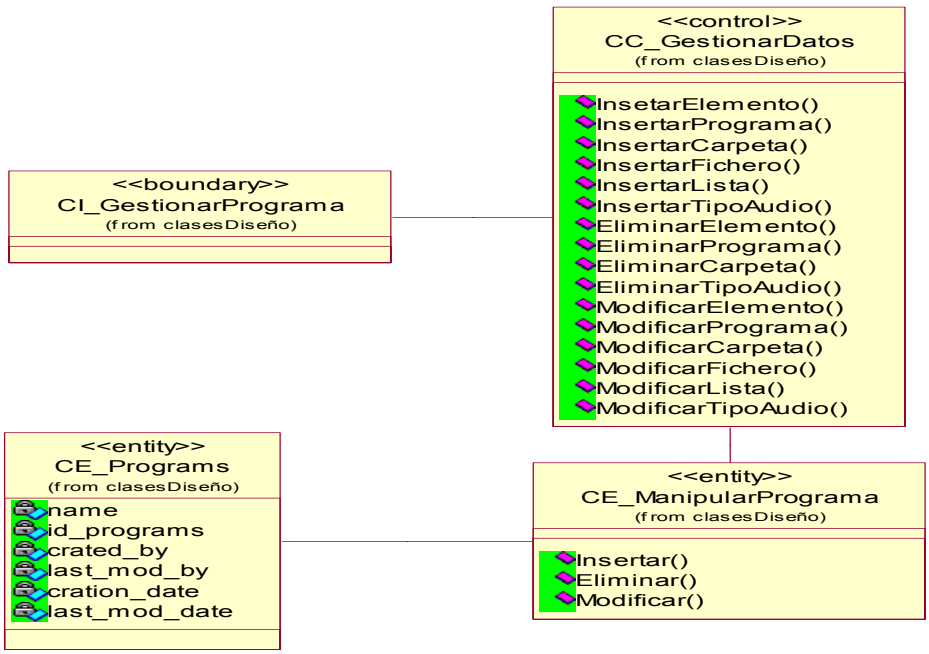
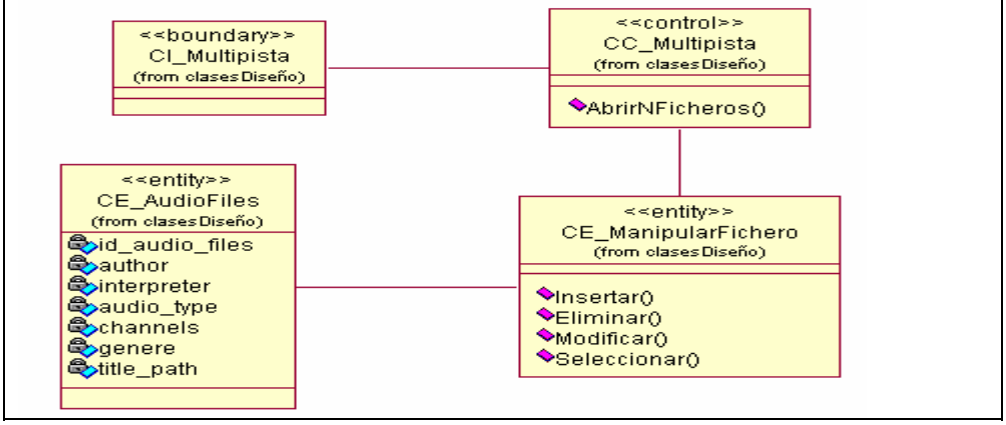
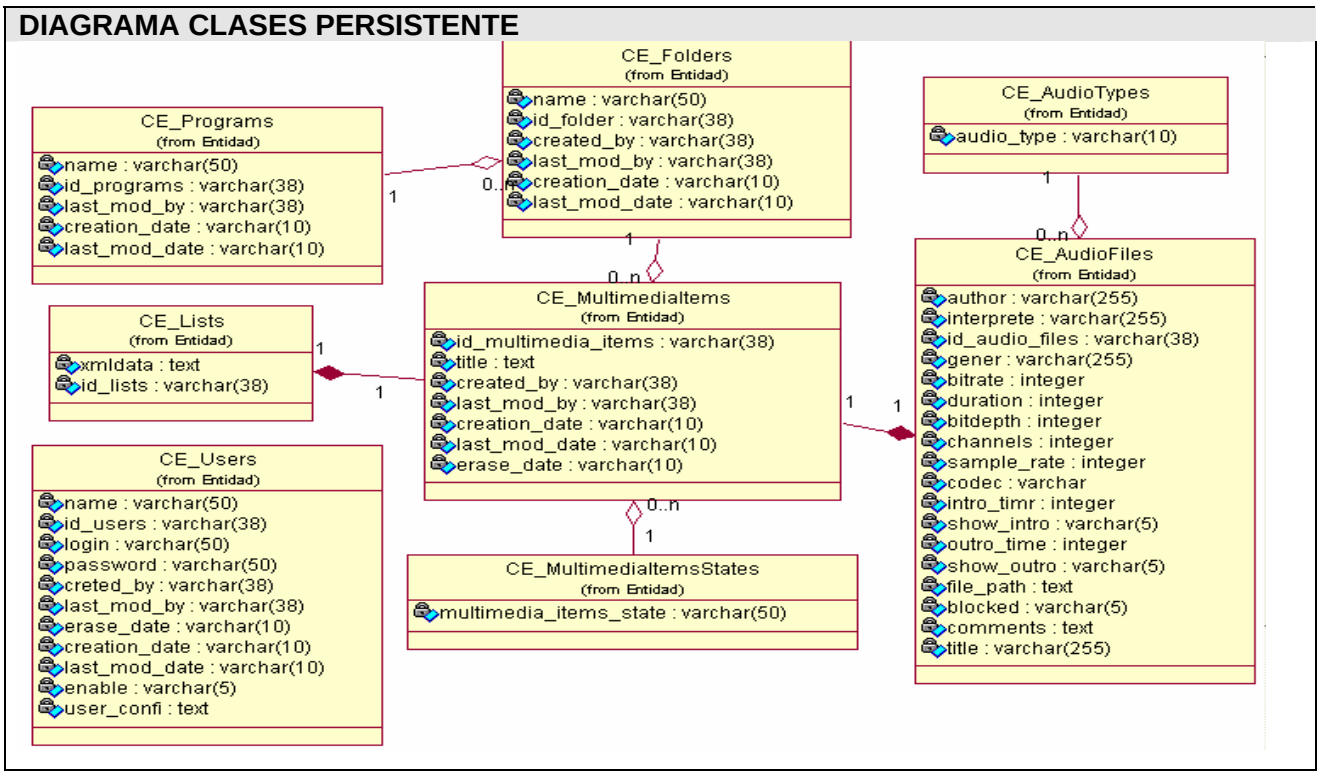


DIAGRAMA DE CLASES CU_Permitir Procesamiento Multipista



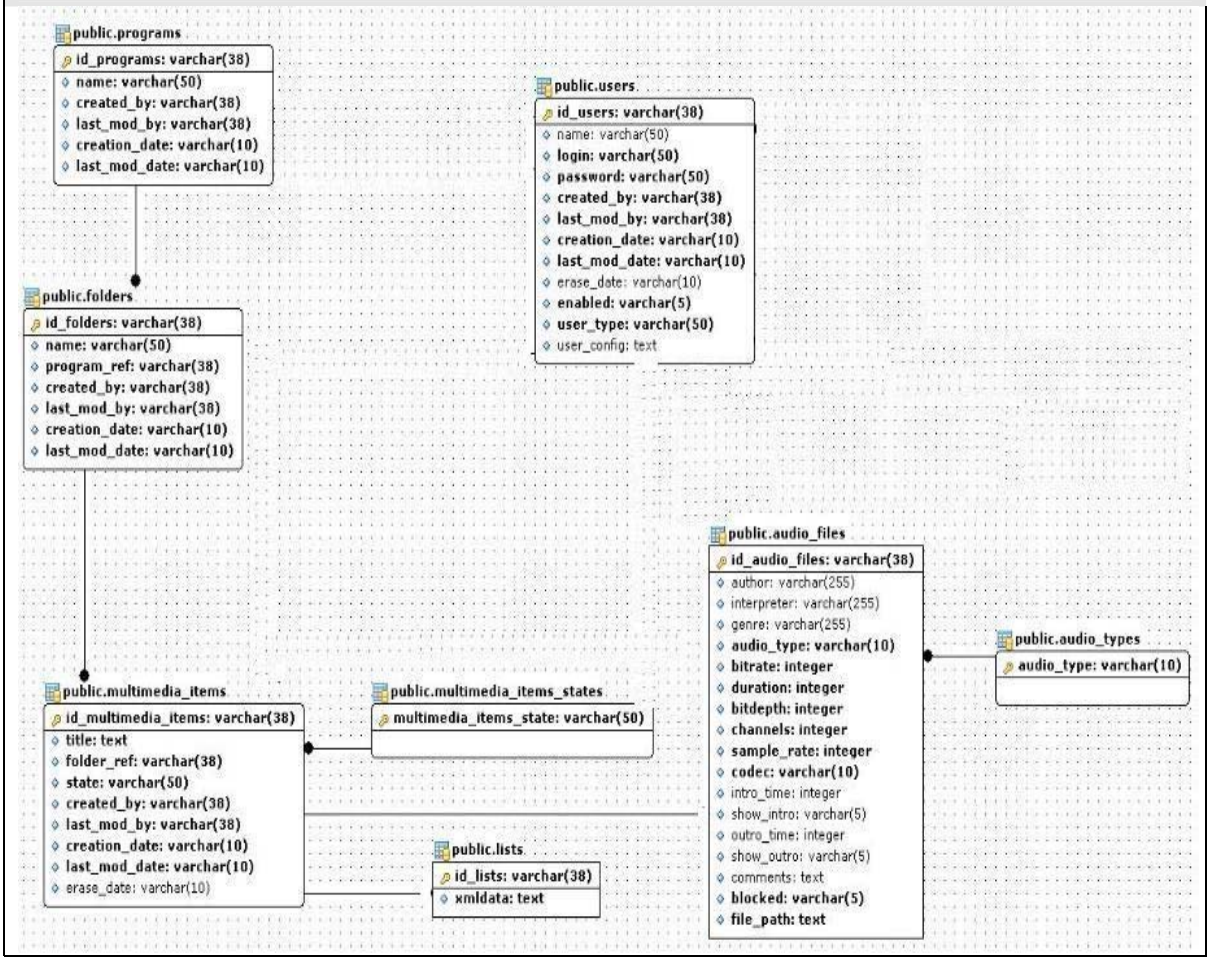
3.4 Diseño de la base de datos

3.4.1 Diagrama de Clases Persistentes



3.4.2 Diagrama entidad relación

DIAGRAMA ENTIDAD RELACION



3.5 Patrones *GRASP* usados

Un conjunto destacado de patrones es GRASP (General Responsabilty Assignment Software Patterns), que permite establecer algunos parámetros útiles para el diseño del producto. Dentro de los patrones GRASP más usados encontramos: Bajo Acoplamiento, Experto, Alta Cohesión, Creador y Controlador. En nuestro sistema usamos el patrón experto y el controlador fundamentalmente. [7]

Experto. La responsabilidad de realizar una labor es de la clase que tiene o puede tener los datos involucrados (atributos). Una clase, contiene toda la información necesaria para realizar la labor que tiene encomendada. Hay que tener en cuenta que esto es aplicable mientras estemos considerando los mismos aspectos del sistema:

- Lógica de negocio
- Persistencia a la base de datos
- Interfaz de usuario

Controlador. Asignar la responsabilidad de controlar el flujo de eventos del sistema, a clases específicas. Esto facilita la centralización de actividades (validaciones, seguridad, etc.). El controlador no realiza estas actividades, las delega en otras clases con las que mantiene un modelo de alta cohesión. Un error muy común es asignarle demasiada responsabilidad y alto nivel de acoplamiento con el resto de los componentes del sistema.

3.6 Patrón *Modelo-Vista-Controlador*

Para el diseño de aplicaciones con sofisticadas interfaces se utiliza el patrón de diseño Modelo-Vista-Controlador. La lógica de un interfaz de usuario cambia con más frecuencia que el almacenamiento 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, traerá como consecuencia modificar trabajosamente los componentes de negocio. Esto trae consigo mayor trabajo y más riesgo de error.

Se trata de realizar un diseño que 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.

Elementos del patrón:

- Modelo: datos y reglas de negocio
- Vista: muestra la información del modelo al usuario
- Controlador: gestiona las entradas del usuario

Un modelo puede tener diversas vistas, cada una con su correspondiente controlador.

Componentes:

1. El modelo es el responsable de:
    - Acceder a la capa de almacenamiento de datos. Lo ideal es que el modelo sea independiente del sistema de almacenamiento.
    - Definir las reglas del negocio (la funcionalidad del sistema). Lleva un registro de las vistas y controladores del sistema.
    - Si se encuentra ante un modelo activo, notificará a las vistas los cambios que en los datos pueda producir un agente externo.
2. El controlador es responsable de:
    - Recibir los eventos de entrada.
    - Contener reglas de gestión de eventos, del tipo "SI Evento Z, entonces Acción W". Estas acciones pueden suponer peticiones al modelo o a las vistas.
3. Las vistas son responsables de:
    - Recibir datos del modelo y mostrarlo al usuario.
    - Tienen un registro de su controlador asociado (normalmente porque además lo instancia).

- Pueden dar el servicio de "Actualización ()", para que sea invocado por el controlador o por el modelo (cuando es un modelo activo que informa de los cambios en los datos producidos por otros agentes).

3.7 Conclusiones

En este capítulo se presentó el análisis y diseño del sistema, los diagramas que fueron necesarios realizar con la ayuda del Rational Rose para obtener una mejor visión de la solución del problema presentado; además se presentó el diseño de la base de datos en la cual se van a almacenar los datos persistentes, además de los patrones GRASP usados y el patrón MVC (Modelo Vista Controlador).

Capítulo 4. Estudio de factibilidad

4.1 Introducción

Para la realización de una aplicación es muy importante hacer un análisis del costo de la misma, por tanto en el presente capítulo haremos un estudio sobre la factibilidad de nuestra aplicación .Para ello aplicaremos la técnica de estimación por caso de uso que nos servirá para calcular el costo, tiempo de desarrollo, el esfuerzo y la cantidad de personas que se necesitan para desarrollar el sistema.

4.2 Puntos de Casos de Uso

1. Cálculo de Puntos de Casos de Uso sin ajustar.

Se calcula a partir de la siguiente ecuación:

UUCP = UAW + UUCW

donde,

UUCP: Puntos de Casos de Uso sin ajustar

UAW: Factor de Peso de los Actores sin ajustar

UUCW: Factor de Peso de los Casos de Uso sin ajustar

2. Factor de Peso de los Actores sin ajustar (UAW)

Los criterios se muestran en la siguiente tabla:

Tabla 15: Peso según tipo de actor

Tipo de Actor	Descripción	Factor de Peso
Simple	Otro sistema que interactúa con el sistema a desarrollar mediante una interfaz de programación (API, Application Programming Interface)	1
Medio	Otro sistema que interactúa con el sistema a desarrollar mediante un protocolo o una interfaz basada en texto	2
Complejo	Una persona que interactúa con el sistema mediante una interfaz gráfica	3

El editor es un tipo de actor complejo, ya que es una persona que interactúa con el sistema mediante una interfaz gráfica, por tanto el factor de peso es igual a:

UAW=1*3=3

3. Factor de Peso de los Casos de Uso sin ajustar (UUCW)

Este valor se calcula mediante un análisis de la cantidad de Casos de Uso presentes en el sistema y la complejidad de cada uno de ellos. La complejidad de los Casos de Uso se establece teniendo en cuenta la cantidad de transacciones efectuadas en el mismo, donde una transacción se entiende como una secuencia de actividades atómica, es decir, se efectúa la secuencia de actividades completa, o no se efectúa ninguna de las actividades de la secuencia. Los criterios se muestran en la siguiente tabla:

Tabla 16: Peso según tipo de caso de uso

Tipo de Caso de Uso	Descripción	Factor de Peso
Simple	El Caso de Uso contiene de 1 a 3 transacciones	5
Medio	El Caso de Uso contiene de 4 a 7 transacciones	10
Complejo	El Caso de Uso contiene más de 8 transacciones	15

Cada uno de los casos de uso “Gestionar Formato”, “Gestionar Elemento”,“Gestionar Programa”, “Gestionar lista”, “Gestionar Carpeta” y “Añadir Efecto” consisten de mas de 8 Transacción. Los casos de uso “Autenticar usuario” y “Multipista” consisten en más de 4 transiciones.

Se tienen entonces 6 casos de uso tipo complejo (peso 15) y 2 casos de uso tipo medio (peso 10), con lo cual el factor de peso de los casos de uso sin ajustar resulta:

UUCW = 6 x1 5 + 2x10 = 90 + 20 = 110

Finalmente, los Puntos de Casos de Uso sin ajustar resultan

UUCP = UAW + UUCW = 3 + 110 = 113

4.3 Cálculo de Puntos de Casos de Uso ajustados

Una vez que se tienen los Puntos de Casos de Uso sin ajustar, se debe ajustar éste valor mediante la siguiente ecuación:

UCP = UUCP x TCF x EF = 113 x 1.005 x 1.115 = 126.62

donde,

UCP: Puntos de Casos de Uso ajustados

UUCP: Puntos de Casos de Uso sin ajustar

TCF: Factor de complejidad técnica

EF: Factor de ambiente

Factor de complejidad técnica (TCF)

Este coeficiente se calcula mediante la cuantificación de un conjunto de factores que determinan la complejidad técnica del sistema. Cada uno de los factores se cuantifica con un valor de 0 a 5, donde 0 significa un aporte irrelevante y 5 un aporte muy importante. En la tabla 17 se muestra el significado y el peso de cada uno de éstos factores:

Tabla 17: Peso según factor

Factor	Descripción	Peso
T1	Sistema distribuido	2
T2	Objetivos de performance o tiempo de respuesta	1
T3	Eficiencia del usuario final	1
T4	Procesamiento interno complejo	1
T5	El código debe ser reutilizable	1
T6	Facilidad de instalación	0.5
T7	Facilidad de uso	0.5
T8	Portabilidad	2
T9	Facilidad de cambio	1
T10	Concurrencia	1
T11	Incluye objetivos especiales de seguridad	1
T12	Provee acceso directo a terceras partes	1
T13	Se requieren facilidades especiales de entrenamiento a usuarios	1

El Factor de complejidad técnica se calcula mediante la siguiente ecuación:

TCF = 0.6 + 0.01 x Σ (Pesoi x Valor asignadoi) = 0.6+0.01 x 40.5 = 1.005

T1- Peso = 2, VA = 5

T2- Peso = 1, VA = 0

T3- Peso = 1, VA = 4

T4- Peso = 1, VA = 5

T5- Peso = 1, VA = 0

T6- Peso = 0.5, VA = 4

T7- Peso = 0.5, VA = 3

T8- Peso = 2, VA = 5

T9- Peso = 1, VA = 4

T10- Peso = 1, VA = 4

T11- Peso = 1, VA = 0

T12- Peso = 1, VA = 0

T13- Peso = 1, VA = 0

Factor de ambiente (EF)

En la siguiente tabla se muestra el significado y el peso de cada uno de éstos factores.

Factor	Descripción	Peso
E1	Familiaridad con el modelo de proyecto utilizado	1.5
E2	Experiencia en la aplicación	0.5
E3	Experiencia en orientación a objetos	1
E4	Capacidad del analista líder	0.5
E5	Motivación	1
E6	Estabilidad de los requerimientos	2
E7	Personal part-time	-1
E8	Dificultad del lenguaje de programación	-1

Para los factores E1 al E4, un valor asignado de 0 significa sin experiencia, 3 experiencia media y 5 amplia experiencia (experto).

El Factor de ambiente se calcula mediante la siguiente ecuación:

EF =1.4 - 0.03 x Σ (Pesoi x Valor asignadoi) = 1.4 – 0.03 x 9.5 = 1.115

De los Puntos de Casos de Uso a la estimación del esfuerzo

Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por debajo del valor medio (3), para los factores E1 a E6.

Se contabilizan cuántos factores de los que afectan al Factor de ambiente están por encima del valor medio (3), para los factores E7 y E8.

- Si el total es 2 o menos, se utiliza el factor de conversión 20 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 20 horas-hombre.

- Si el total es 3 o 4, se utiliza el factor de conversión 28 horas-hombre/Punto de Casos de Uso, es decir, un Punto de Caso de Uso toma 28 horas-hombre.

- Si el total es mayor o igual que 5, se recomienda efectuar cambios en el proyecto, ya que se considera que el riesgo de fracaso del mismo es demasiado alto.

CF = 28 horas-hombre/Punto de Casos de Uso

E1- Peso = 1.5, VA = 0

E2- Peso = 0.5, VA = 0

E3- Peso = 1, VA = 3

E4- Peso = 0.5, VA = 3

E5- Peso = 1, VA = 5

E6- Peso = 2, VA = 0

E7- Peso = -1, VA = 0

E8- Peso = -1, VA = 0

El esfuerzo en horas-hombre viene dado por:

E = UCP x CF = 126.62 X 28 = 3545.36 horas-hombre

Donde

E: esfuerzo estimado en horas-hombre

UCP: Puntos de Casos de Uso ajustados

CF: factor de conversión

Este método proporciona una estimación del esfuerzo en horas-hombre contemplando sólo el desarrollo de la funcionalidad especificada en los casos de uso.

Asumimos que vamos a trabajar 8 horas diarias durante 22 días, al mes serían 176 horas.

Tabla 18: Esfuerzo por actividad

Actividades	**% Esfuerzo**	**Valor Esfuerzo (horas-hombre)**
Análisis	10	886.34
Diseño	20	1772.68
Implementación	40	3545.36
Prueba	15	1329.51
Sobre carga	15	1329.51
Total	100	8863.4

Et = Ea + Ed+ Ei + Ep + Esc = 886.34 + 1772.68 + 3545.36 + 1329.51 + 1329.51

Et = 8863.4 horas-hombre = 50.4 mes – hombre.

Et: esfuerzo total

TD: tiempo de desarrollo

CH: cantidad de hombres

SM: salario medio

CH = 8 hombres

SM = $ 75

TD = Et / CH

TD = 50.4 / 8 = 6 meses

C = TD x CH x SM

C = 6 x 8 x 75 =$ 3600

4.4 Cálculos del Proyecto

Tabla 19: Resultados del proyecto

Cálculo de	**Valor**
Esfuerzo	50.4 mes – hombre
Tiempo de desarrollo	6 meses
Cantidad de hombres	8
Salario medio	$ 75
Costo	$ 3600

4.4.1 Para el análisis y diseño

Tabla 20: Resultados para el análisis y diseño

Cálculo de	Valor
Esfuerzo	15.1 mes – hombre
Tiempo de desarrollo	2 meses
Cantidad de hombres	8
Salario medio	$ 75
Costo	$1200

4.5 Beneficios tangibles e intangibles. Debe cuantificarse los tangibles.

El Sistema tiene como objetivo principal solucionar el problema al que se enfrentan los editores de la radio cubana, cuando tienen que cargar un fichero o salvar una edición.

El beneficio fundamental de este sistema es permitir que el editor pueda cargar los ficheros de la base de datos, salvar las ediciones, organizar mejor su trabajo, con la organización en programas y carpetas que va a tener en la base de datos, lo que le va a permitir hacer su trabajo en el menor tiempo.

Por tanto, podemos plantear los siguientes beneficios:

- Disminución del tiempo y esfuerzo que invierten los editores a la hora de editar.
- Fácil y rápido acceso a la información actualizada en la bases de datos.

4.6 Análisis de costo

Antes de desarrollar una aplicación es muy importante que primero hagamos un análisis de su costo ,ya que este nos dirá si los beneficios que ella nos reportara se corresponden con su costo y de acuerdo a esto se considerara si se continua o no con su desarrollo.

Estudio de factibilidad.

El sistema que se propone está dirigido fundamentalmente a apoyar el trabajo de edición en la radio cubana. Una vez implementado el sistema contribuirá en gran mediada con el trabajo de los editores.

Analizando el costo del proyecto, los numerosos beneficios que reporta, detallados con anterioridad, se puede concluir que su implementación es realmente factible.

4.7 Conclusiones

En este capítulo se describió el estudio de factibilidad realizado al sistema propuesto a través de la técnica de puntos de casos de uso, mediante el cual se obtuvo el costo y tiempo de desarrollo, cantidad de personas, es decir datos con los que podemos medir la viabilidad del proyecto. Se analizaron además los beneficios que este nos reportará al ser implantado.

Por todo esto se llego a la conclusión que es factible implementar el sistema propuesto

Conclusiones Generales

Se decidió que el Sistema se implementará utilizando la plataforma .NET. El lenguaje de programación C#. La metodología RUP, utilizando UML como lenguaje de modelación. Como gestor de base de datos PostgreSQL 8.1. Para describir el contexto en que se desenvuelve se realiza el modelo del dominio. Se definieron los requisitos funcionales y no funcionales que debe cumplir el sistema. Se definieron los patrones GRASP y arquitectónico, que se utilizarán. Se llegó a la conclusión que el proyecto es factible.

Tabla 17: Resultados del proyecto

Cálculo de	**Valor**
Esfuerzo	50.4 mes – hombre
Tiempo de desarrollo	6 meses
Cantidad de hombres	8
Salario medio	$ 75
Costo	$ 3600

Recomendaciones

Se recomienda que se termine la implementación de este sistema, el mismo estará dirigido fundamentalmente a apoyar el trabajo de edición en la radio cubana, lo cual tributará a un incremento de la eficiencia y calidad de las emisiones radiales en Cuba, además de un ahorro considerable de divisas, por concepto de licencias y de knowhow.

Glosario de términos

- CU: caso de uso.
- RUP: Proceso Unificado de Modelado.
- RF: requerimientos funcionales.
- RNF: requerimientos no funcionales.
- BD: base de datos.
- MVC: modelo vista controlador.

Diagramas de interacción

Referencias

[1] Equipo de Softonic. Sound Forge, 2007 [http://sound-forge.softonic.com/]

[2] Julián Gómez. Audacity, 2007 [http://audacity.softonic.com/]

[3] The Sonic Spot. Cool Edit, 2007 [http://www.sonicspot.com/cooledit/cooledit.html]

[4] Equipo de Softonic. Fleximusic Wave Editor, 2006 [http://fleximusic-wave- editor.softonic.com/]

[5] Digigram. Xtrack Audio Suite, 2007 [http://www.digigram.com/products/getinfo.htm?prod_key=9050]

[6] Ivar Jacobson, Grady Booch, James Rumbaugh. El proceso unificado de desarrollo de software, Volumen I. La Habana, Editorial Félix Varela, 2004. pp 4 – 12

[7] Craig Larman. UML y patrones, Volumen I. La Habana, Editorial Félix Varela, 2004. pp 3 – 6, pp 193 – 210

[8] Profesores del departamento de ISW, Introducción a la Ingeniería de Software, UCI, 2005

[9] IBM Rational Rose Enterprise. Rational Rose Enterprise, 2007. [http://www-950.ibm.com/ecatalog/Detail.wss?locale=es_ES&synkey=M221280M46834Z27&&S_TACT=none&S_CMP=none]

[10] Javier Quiroz. El modelo relacional de bases de datos, 2003 [http://www.inegi.gob.mx/inegi/contenidos/espanol/prensa/Contenidos/Articulos/tecnologia/relacional.pdf]

[11] Wikipedia. PostgreSQL, 2007 [http://es.wikipedia.org/wiki/PostgreSQL]

[12] Wikipedia. .Net, 2006 [http://es.wikipedia.org/wiki/.NET]

[13]Wikipedia. Visual Studio 2005, 2007. [http://es.wikipedia.org/wiki/visual_estudio]

[14] Wikipedia. SharpDevelop, 2007 [http://es.wikipedia.org/wiki/SharpDevelop]

[15] José Antonio González Seco. El lenguaje de programación C#, 2007.
[http://www.programacion.net/tutorial/csharp/3/]
[16] Álvarez, Sofía, Hernández Anaisa. Metodología para el desarrollo de aplicaciones con tecnología Orientada a Objetos utilizando notación UML. La Habana, 2000

Bibliografía

- Álvarez, Sofía, Hernández Anaisa. Metodología para el desarrollo de aplicaciones con tecnología Orientada a Objetos utilizando notación UML. La Habana, 2000
- Craig Larman. UML y patrones, Volumen I. La Habana, Editorial Félix Varela, 2004. pp 3 – 6, pp 193 – 210
- Digigram. Xtrack Audio Suite, 2007[http://www.digigram.com/products/getinfo.htm?prod_key=9050]
- Equipo de Softonic. Fleximusic Wave Editor, 2006 [http://fleximusic-wave- editor.softonic.com/]
- Equipo de Softonic. Sound Forge, 2007 [http://sound-forge.softonic.com/]
- IBM Rational Rose Enterprise. Rational Rose Enterprise, 2007. [http://www-950.ibm.com/ecatalog/Detail.wss?locale=es_ES&synkey=M221280M46834Z27&&S_TACT=none&S_CMP=none]
- Ivar Jacobson, Grady Booch, James Rumbaugh. El proceso unificado de desarrollo de software, Volumen I. La Habana, Editorial Félix Varela, 2004. pp 4 – 12
- JAMES RUMBAUGH, I. J., GRADY BOOCH. *El lenguaje unificado de modelado. Manual de Referencia*, 2001. [http://bibliodoc.uci.cu/pdf/reg03050.pdf]

- José Antonio González Seco. El lenguaje de programación C#, 2007. [http://www.programacion.net/tutorial/csharp/3/] Julián Gómez. Audacity, 2007 [http://audacity.softonic.com/]
- Profesores del departamento de ISW, Introducción a la Ingeniería de Software, UCI, 2005
- The Sonic Spot. Cool Edit, 2007 [http://www.sonicspot.com/cooledit/cooledit.html]
- Wikipedia. .Net, 2006 [http://es.wikipedia.org/wiki/.NET]
- Wikipedia. MVC, 2007 [http://es.wikipedia.org/wiki/Modelo_Vista_Controlador]
- Wikipedia. PostgreSQL, 2007 [http://es.wikipedia.org/wiki/PostgreSQL]
- Wikipedia. SharpDevelop, 2007 [http://es.wikipedia.org/wiki/SharpDevelop]
- Wikipedia. Visual Studio 2005, 2007. [http://es.wikipedia.org/wiki/visual_estudio]

DIAGRAMA DE INTERACCION CU_AñadirEfecto

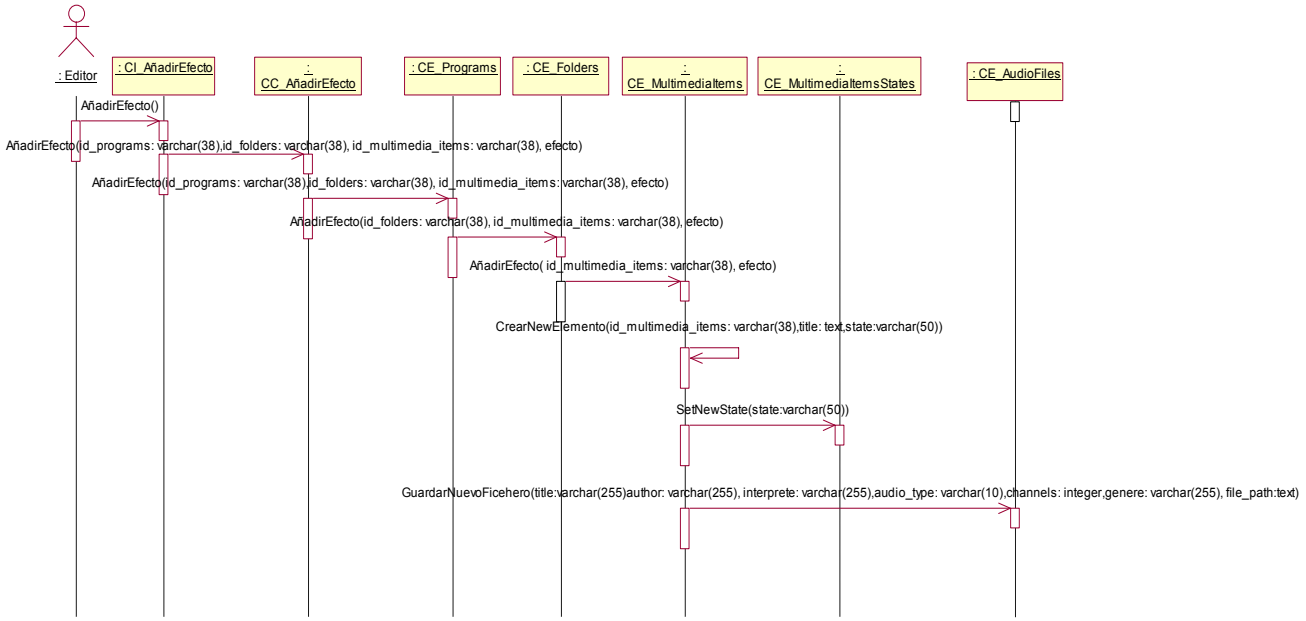


DIAGRAMA DE INTERACCION CU_Autenticar usuario

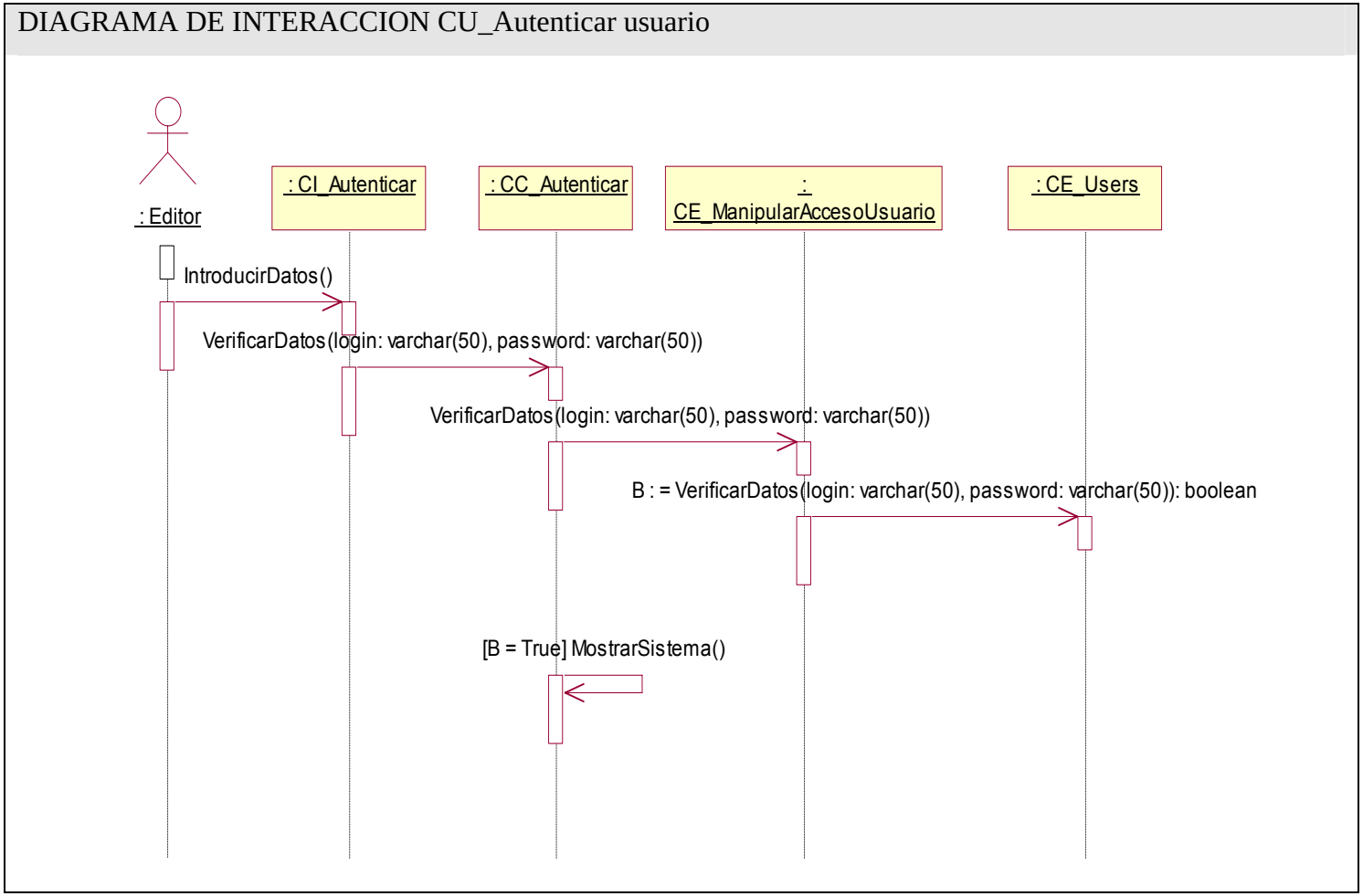
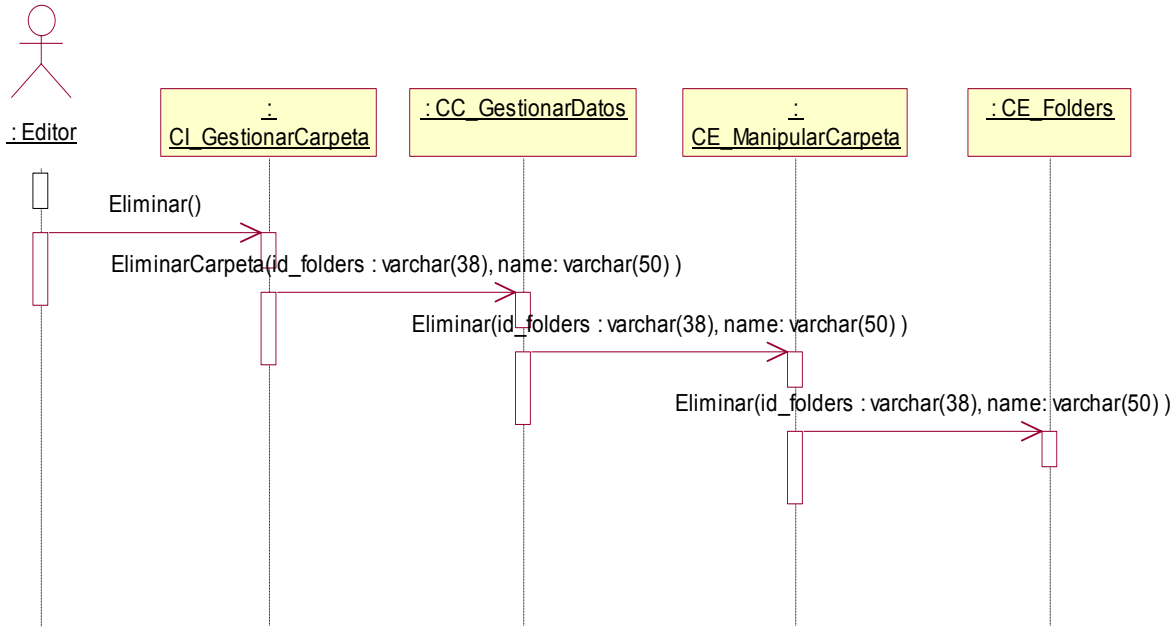


DIAGRAMA DE INTERACCION CU_GestionarCarpeta(Escenario Eliminar)



Diagramas de interacción

DIAGRAMA DE INTERACCION CU_GestionarCarpeta(Escenario Insertar)

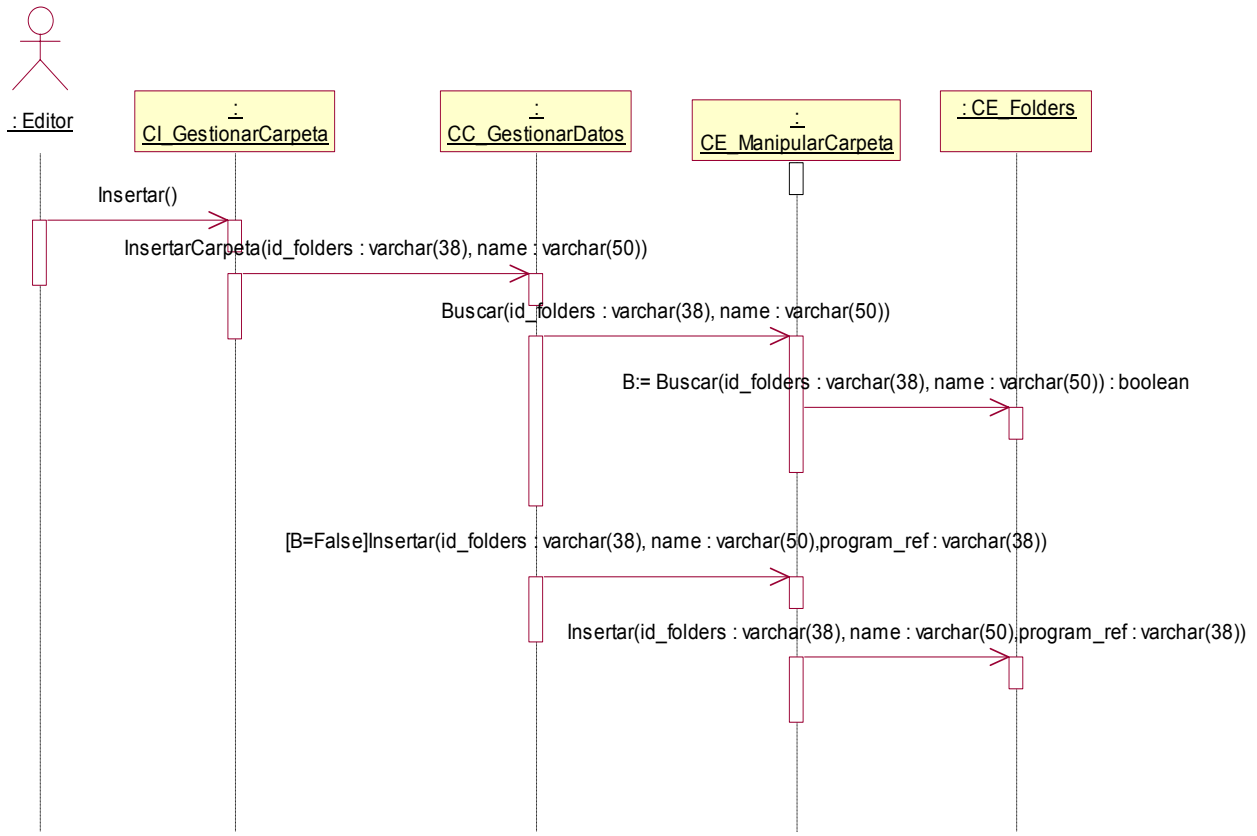


DIAGRAMA DE INTERACCION CU_GestionarCarpeta(Escenario Modificar)

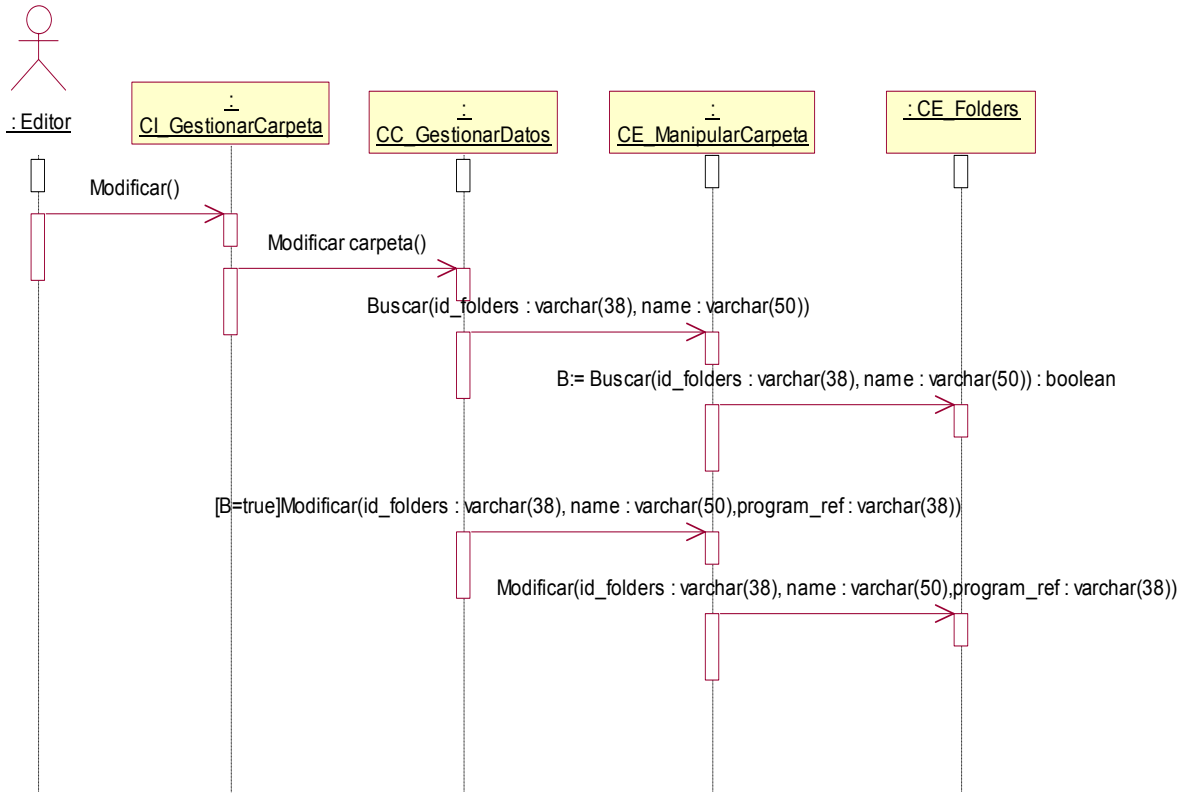
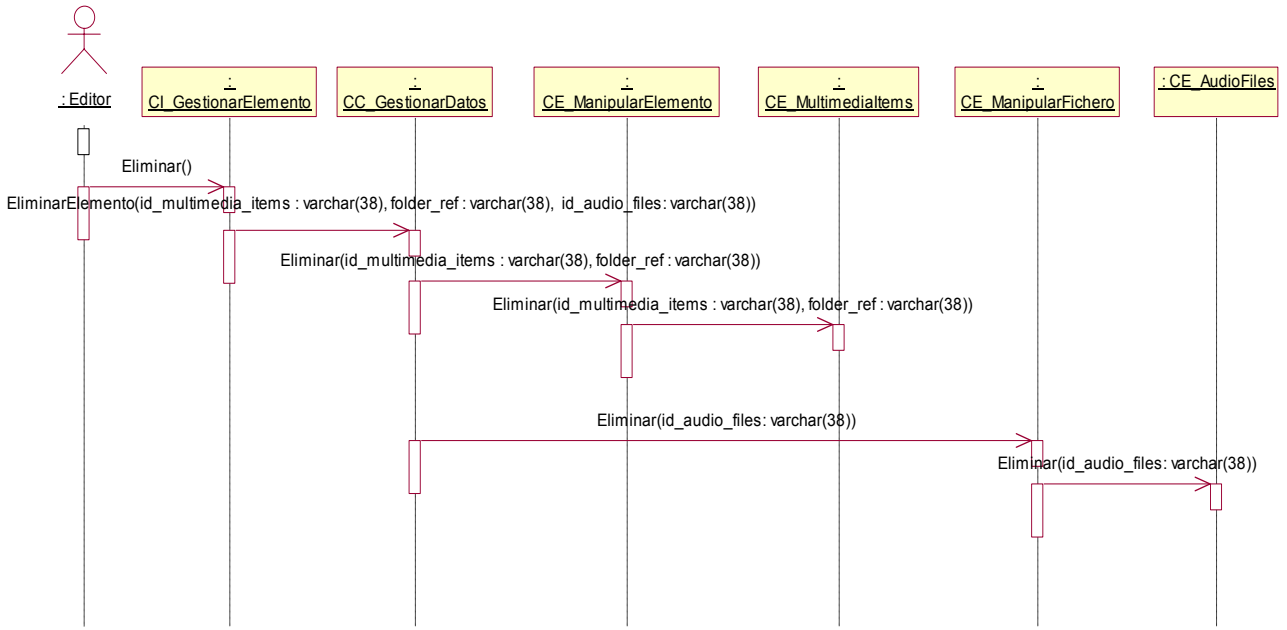


DIAGRAMA DE INTERACCION CU_GestionarElemento(Escenario Eliminar)



Diagramas de interacción

DIAGRAMA DE INTERACCION CU_GestionarElemento(Escenario Insertar)

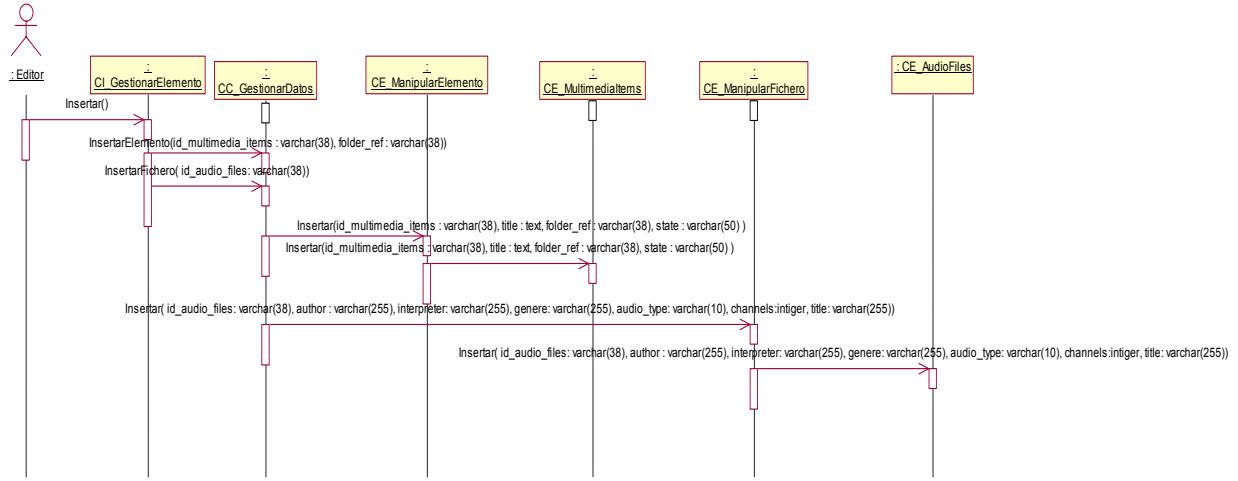


DIAGRAMA DE INTERACCION CU_GestionarElemento(Escenario Modificar)

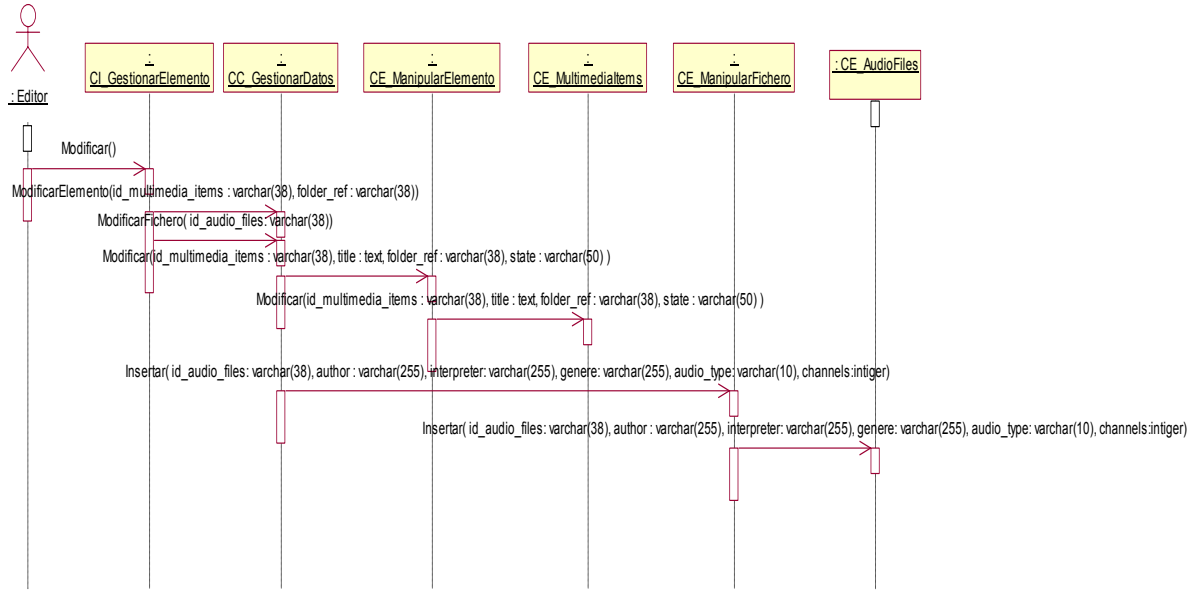


DIAGRAMA DE INTERACCION CU_GestionarTipoAudio(Escenario Eliminar)

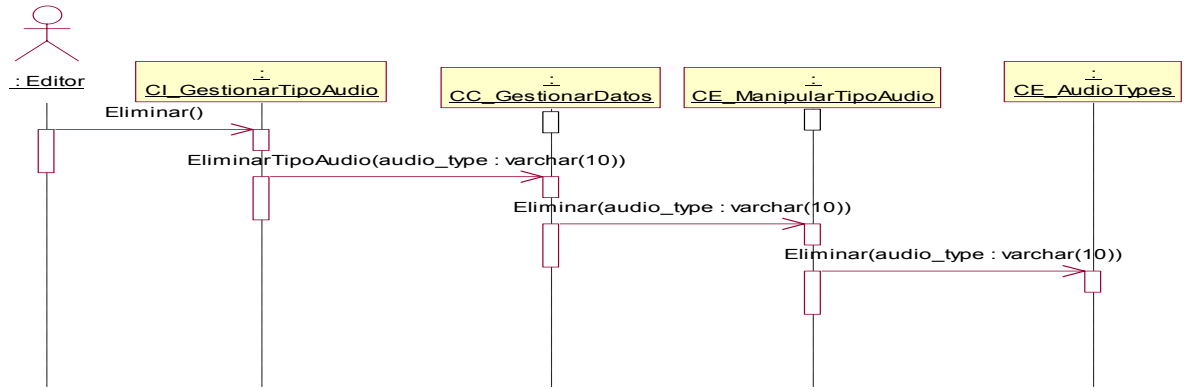


DIAGRAMA DE INTERACCION CU_GestionarTipoAudio(Escenario Insertar)

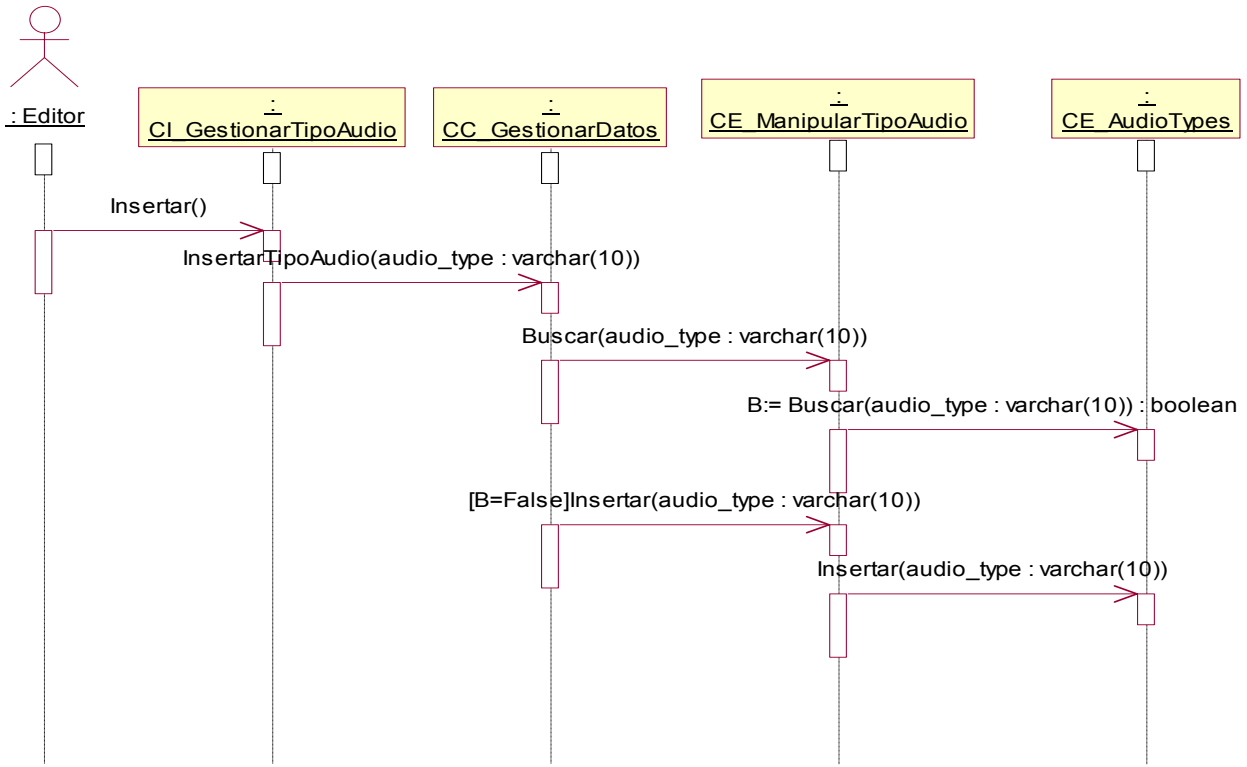
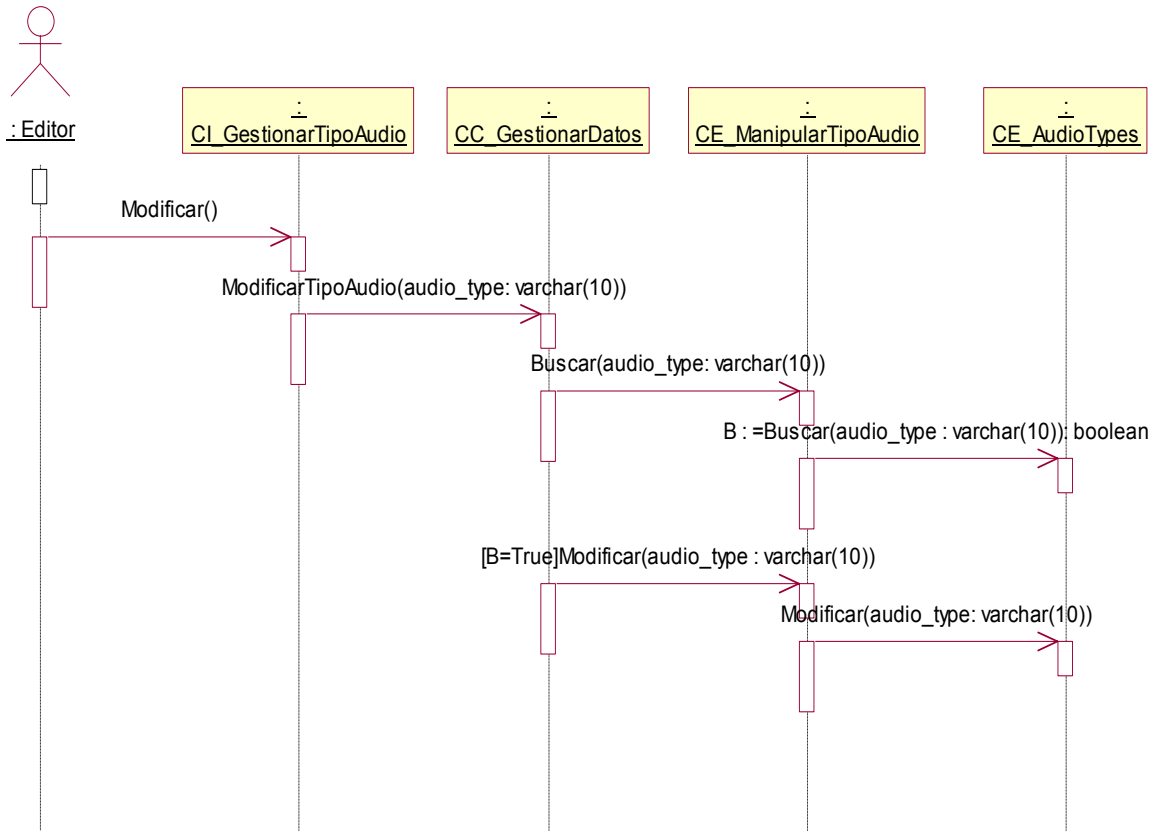


DIAGRAMA DE INTERACCION CU_GestionarTipoAudio(Escenario Modificar)



Diagramas de interacción

DIAGRAMA DE INTERACCION CU_GestionarLista(Escenario Eliminar)

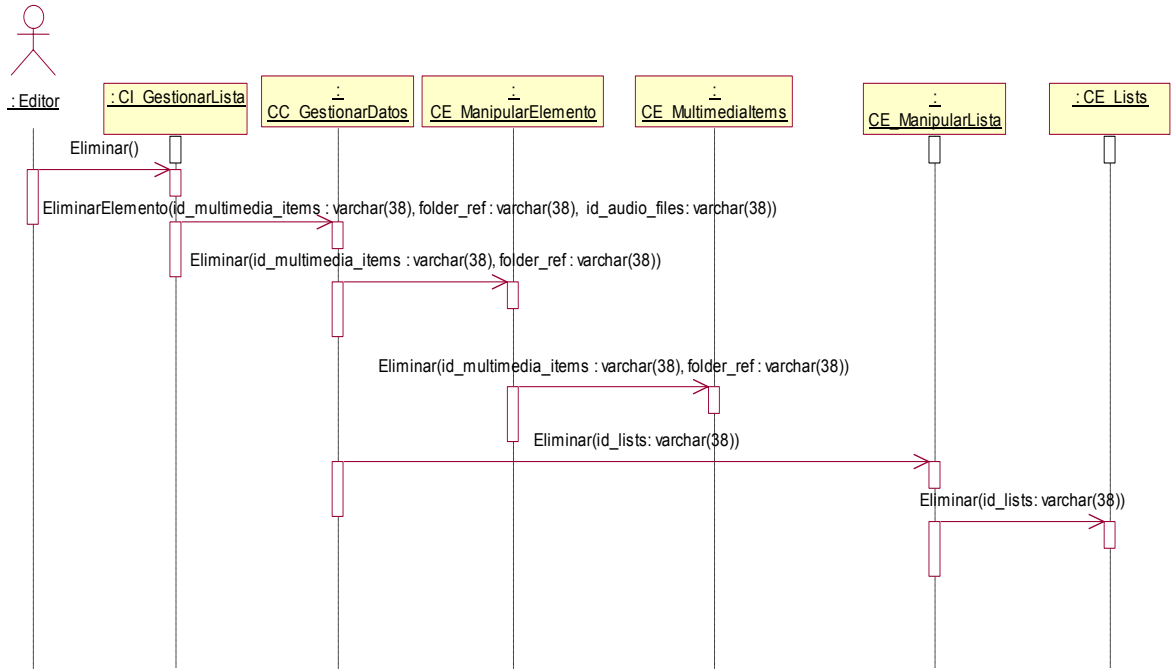
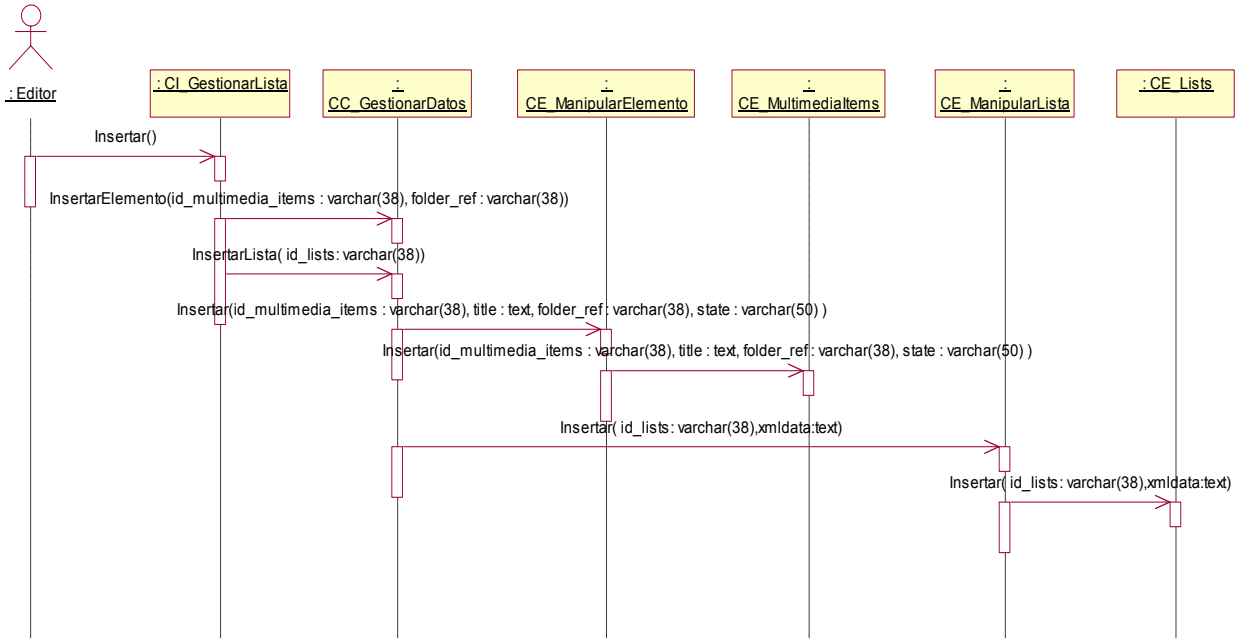
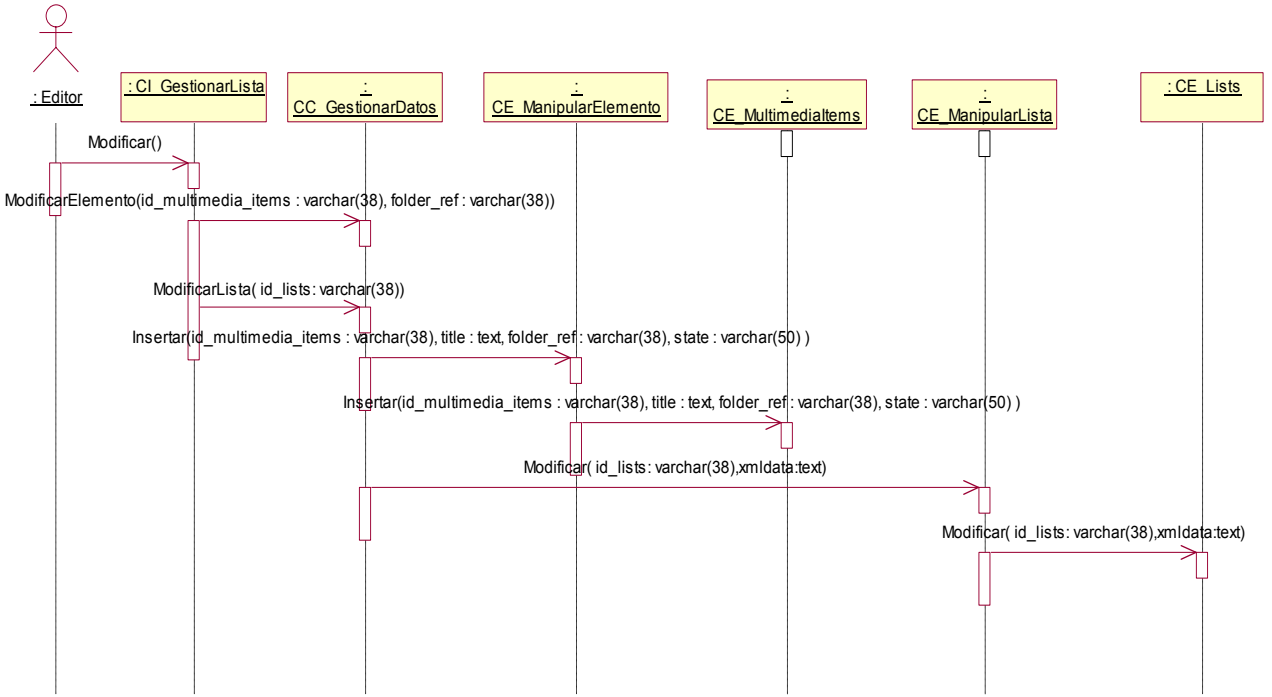


DIAGRAMA DE INTERACCION CU_GestionarLista(Escenario Insertar)



Diagramas de interacción

DIAGRAMA DE INTERACCION CU_GestionarLista(Escenario Modificar)



Diagramas de interacción

DIAGRAMA DE INTERACCION CU_GestionarPrograma(Escenario Eliminar)

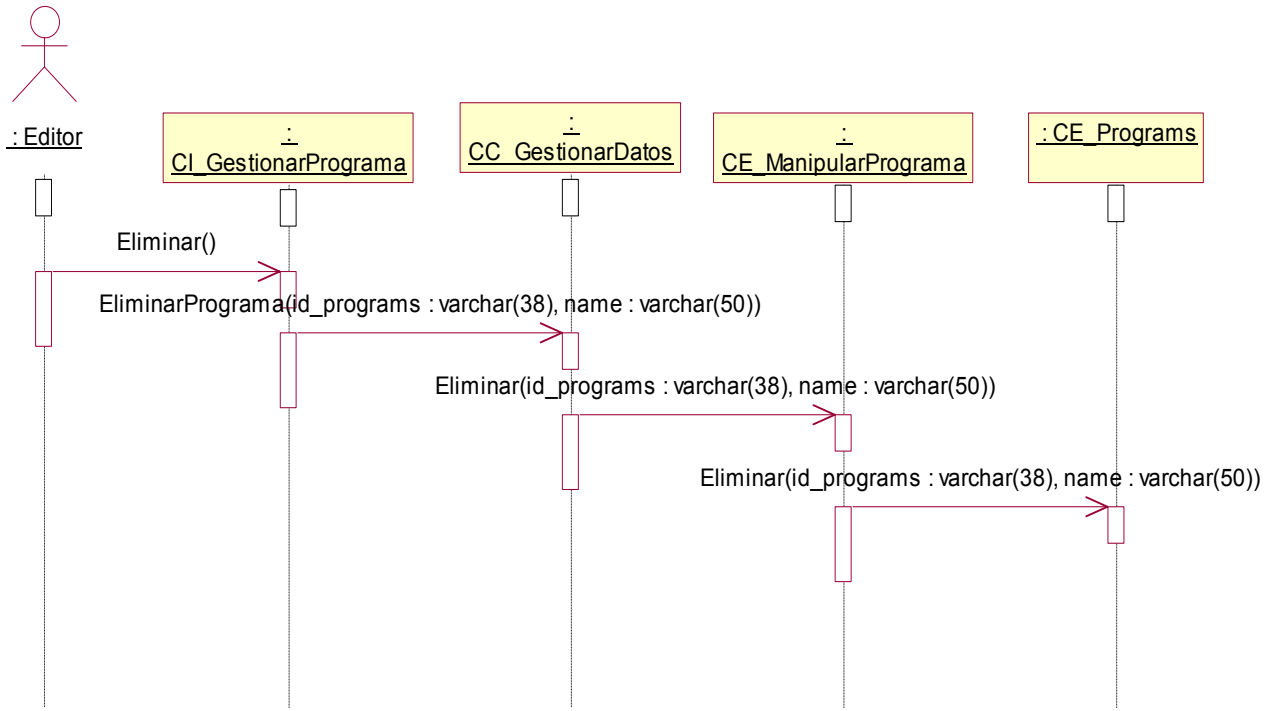


DIAGRAMA DE INTERACCION CU_GestionarPrograma(Escenario Insertar)

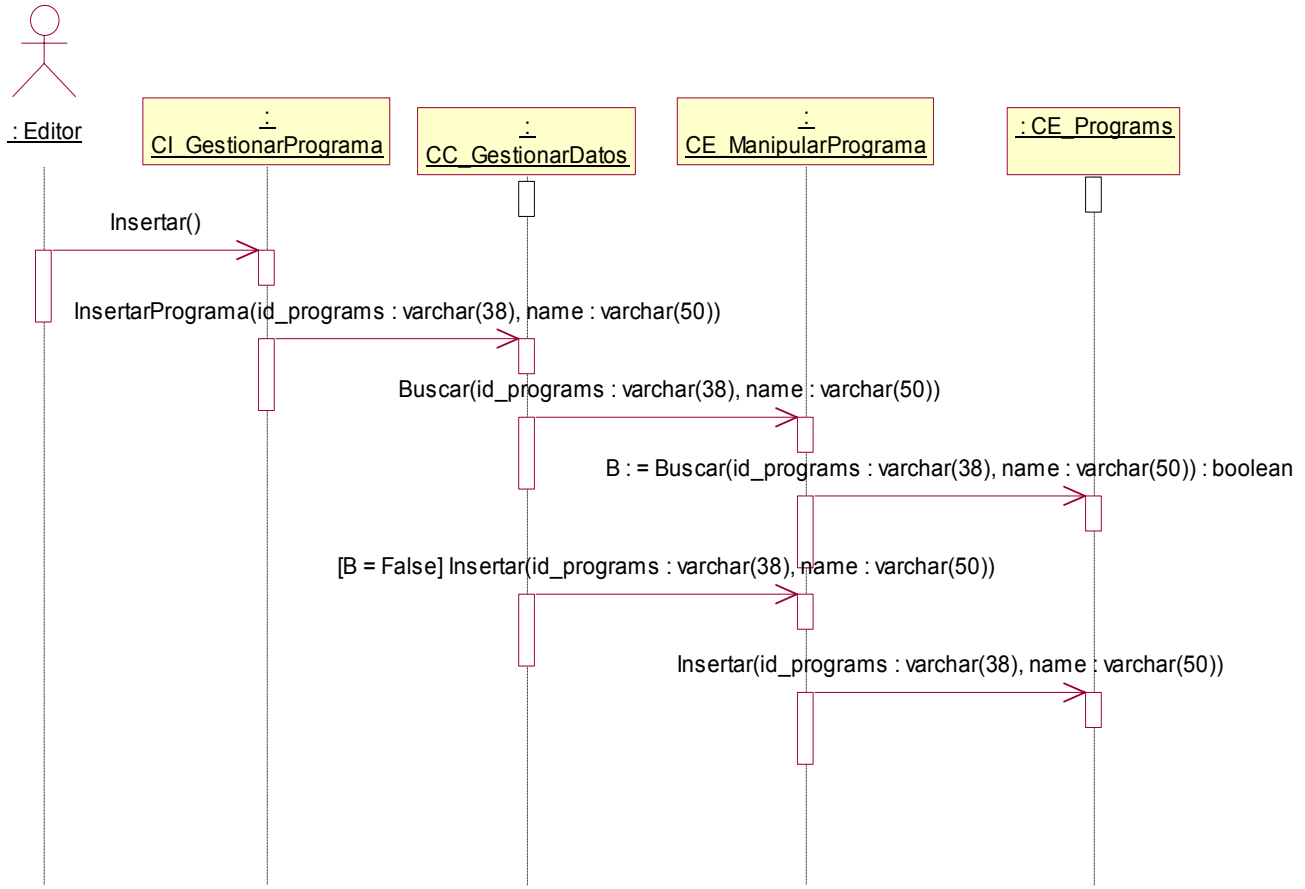


DIAGRAMA DE INTERACCION CU_GestionarPrograma(Escenario Modificar)

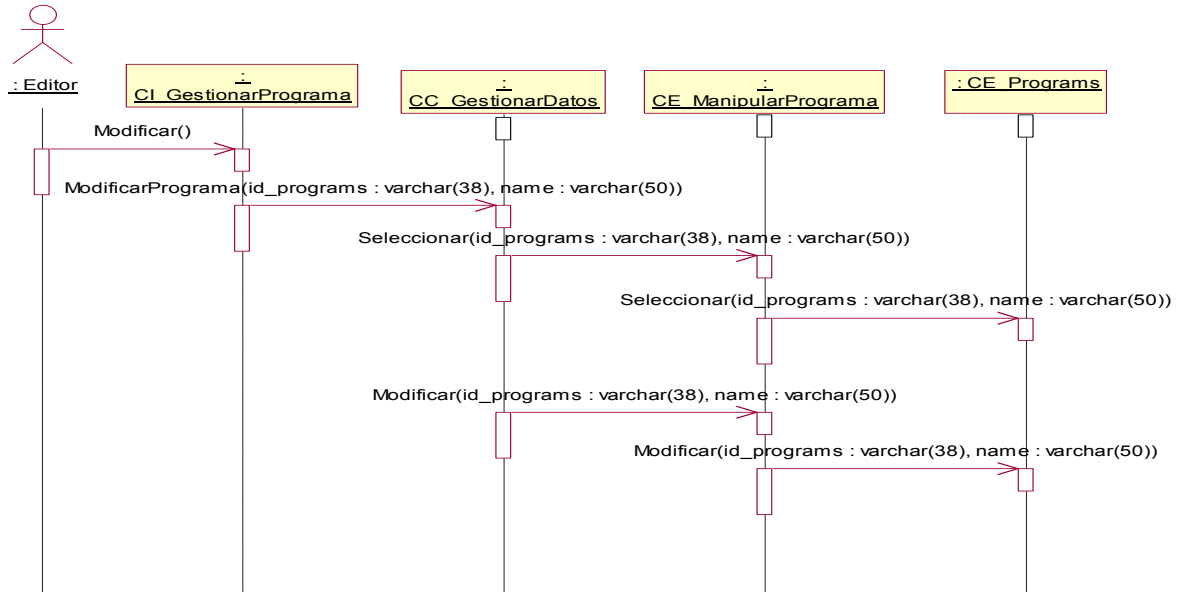


DIAGRAMA DE INTERACCION CU_Permitir Procesamiento Multipista