Universidad de las Ciencias Informáticas

Facultad 10

Título: Diseño e implementación de un categorizador automático de imágenes pornográficas

Trabajo de Diploma para optar por el Título de Ingeniero en Ciencias Informáticas

Autor: Kiuver Kaddiel Ibañez Castro

Tutor: Ing. Siovel Rodríguez Morales

Ciudad de la Habana, 3 de Julio de 2008

“Codifica siempre como si la persona que finalmente mantendrá tu código fuera un psicópata violento que sabe dónde vives”

Martin Golding

Declaración de autoria

Declaro ser autor 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 _____.

Kiuver Kaddiel Ibañez Castro	Ing. Siovel Rodríguez Morales
Autor	Tutor

Agradecimientos

Agradezco primeramente a mi tutor, Siovel, el cual me ha dado la ayuda y guía necesaria para la realización del presente trabajo de diploma, sin el no hubiese sido posible mi graduación. A Dovier por haberme convidado a participar en el proyecto FILPACON lo que representa para mi la mejor de las cosas que me ha sucedido en la universidad. A mis compañeros de proyecto más allegados Dovier, Luis, Jose y Yandry porque entre “chistes”, situaciones cómicas y reflexiones han logrado hacer mi vida más amena y feliz. A mis hermanos de la vocacional a los cuales quiero y estimo por haberme brindado durante muchos años su amistad y ayuda desinteresada e incondicional. A Rolando y Michel mis amigos entrañable de la universidad que nunca olvidaré. A Suri, Indira y Maria mis babys que llenan mi vida de colores y quisiera haberlas conocido antes.

Por ultimo a las personas que me han ayudado a llegar a este punto de mi vida. A mi tía Margarita y mis dos primas Adriana y Lisett por brindarle a mi familia todo el apoyo que ha necesitado. A mi padre, Roberto, por haberme mostrado desde temprana edad mi vacación lo cual ha sido mi guía en todo momento. A mi madre, Sonia, por darme todo el amor que una madre puede dar a un hijo. A mi abuelo, Luis, por ser mi faro en el afán de convertirme en el hombre que debiera ser.

A la Revolución cubana...

A la Universidad de las Ciencias Informáticas...

A todo aquel que me robó una sonrisa...

Dedicatoria

Dedico este trabajo de diploma a toda mi familia especialmente a mi tía Margarita y mis dos primas Adriana y Lisett; a mis padres Reberto y Sonia; hermanos Karlen, Karel, Raúl y Kirenia; y abuelo Luis. Este triunfo de convertirme en ingeniero informático no es mio, es de ustedes; por lo que les deseo muchas felicidades.

Resumen

En la Universidad de las Ciencias Informáticas se está implementando, desde el año 2005 a petición de la Oficina de Seguridad para las Redes Informáticas, un sistema de filtrado que pretende; de forma flexible, regular los contenidos inadecuados que provienen de Internet. Este sistema de filtrado tiene como componente principal una Base de Datos de URLs Categorizadas mediante la cual se toma la decisión de autorizar o denegar el acceso al los recursos solicitados. Dado que es inviable la categorización de las URLs de forma manual se necesita para esto un proceso automático por lo que surge el Motor de Categorización Automatizada de Documentos HTML. Este motor es el encargado de clasificar de forma automática las páginas Web, para poder dar un criterio general de una página Web es necesario la categorización tato de su texto como de sus imágenes. Las imágenes constituyen un componente esencial en las páginas Web, estás son mayormente usadas para hacer más atractivos los contenidos, sin embargo también están contribuyendo a la existencia de contenidos dañinos, que a su ves pueden ser ilegales, tales como la pornografía y la pedofilia. En este artículo se propone un módulo para la categorización automática de imágenes pornográficas con vistas a su aplicación en el Motor de Categorización Automatizada de Documentos HTML. Su aplicación permitirá mejorar la categorización de las páginas Web y sentará las bases para el desarrollo de nuevos métodos más efectivos.

Palabras claves: Reconocimiento de patrones, Segmentación, Redes Neuronales Artificiales

Índice general

Índice de figuras

Índice de cuadros

Introducción

Internet es una sistema de redes distribuidas que brinda diferentes servicios[1]. Gracias a esto cualquiera que tenga acceso a esta puede ofrecer un servicio agregando la capacidad de su ordenador a la gran red de redes. Esto a su ves a propiciado un crecimiento exponencial dada la necesidad de las personas o instituciones de ofrecer y recibir dichos servicios. Internet constituye, hasta la fecha, la via de comunicación más rápida y eficiente de todos los tiempos. Dado su rapidez y su enorme volumen de materiales de todo tipo, es la fuente de información de muchos estudiantes, investigadores, científicos, entre otros.

En el 2006 el universo digital alcanzó la dimensión de 161 exabytes[2], este universo digital equivale aproximadamente a 3 millones de veces la cantidad total de libros escritos en la historia de la humanidad o el equivalente a 12 pilas de libros, cada una ellas extendiéndose a 93 millones de millas desde la Tierra al Sol. En este mismo año, a escala mundial, las imágenes capturadas por cámaras digitales excedieron los 150 mil millones, mientras que las imágenes capturadas por teléfonos celulares alcanzaron la cantidad de 100 mil millones. De igual forma las direcciones de correo electrónico crecieron de 253 millones en el año 1998 a 1.600 millones en el año 2006. En ese lapso, la cantidad de correos electrónicos enviados creció a una tasa tres veces mayor que las personas que utilizan este servicio y el tráfico unilateral de estos, incluyendo el correo basura[3], alcanzó los 6 exabytes[1, 2].

El servicio más difundido de Internet, la WWW, no está exento de este crecimiento, un estudio realizado por netcraft[4] en el presente año arrojó lo existencia de cerca de 170 millones de sitios Web, de los cuales

[1] Los servicios más conocidos de Internet son la WWW, el correo electrónico y el servicio de mensajería instantánea; aunque existen muchos más (http://www.monografias.com/trabajos14/servic-internet/servic-internet.shtml).

[2] Representa mil millones de gigabyte

[3] Lo que se conoce en la bibliografía anglosajona como spam

[4] http://news.netcraft.com/archives/2008/05/06/may_2008_web_server_survey.html

68 millones son sitios activos[5], ver Anexo A.

Lamentablemente no todos los contenidos que se encuentran en Internet poseen valor[6], como es el caso de la pornografía. La tabla 1 muestra alarmantes estadísticas de este tipo de contenido. Las leyes internacionales poco han podido hacer ya que, como se dijo, Internet es una red distribuida por lo que no tiene jurisdicción; por ejemplo en nuestro país la pornografía está penada por la ley mientras que en Estados Unidos es totalmente libre. Hasta ahora solo las soluciones técnicas han sido capases de regular estos contenidos inadecuados. Un ejemplo de esas soluciones técnicas son los sistemas de filtrado.

Sitios pornográficos	12 % del total
Correos diarios con material pornográfico	2.5 mil millones (8 % del total)
Promedio diario de correos pornográfico por usuario	4.5 por usuario de Internet
Descargas mensuales de pornografía (punto a punto)	1.5 mil millones (35 % del total)

Cuadro 1: Estadísticas sobre material pornográfico en Internet

En la Universidad de las Ciencias Informáticas (UCI) se está desarrollando, a petición de la Oficina de Seguridad para las Redes Informáticas (OSRI), un proyecto llamado Filtrado de Paquetes por Contenido (FILPACON) el cual pretende regular el flujo de materiales inadecuados a nuestra red, el funcionamiento general de este filtro se puede ver en el Anexo B. El componente fundamental de FILPACON es una Base de Datos de URLs Categorizadas (BDUC), dado el gran volumen que posee Internet, llenar esta de forma manual es una tarea inviable.

Para mantener actualizada la BDUC se usará el Motor de Categorización Automatizada de Documentos HTML (MCADHTML), el cual se encuentra en desarrollo. Uno de los módulos identificados en este motor es el módulo para la categorización de imágenes pornográficas, el cual es fundamental para la categorización general del documento HTML dado que las imágenes constituyen un componente importante en las páginas Web, aún más en las páginas Web que tienen contenidos pornográficos. La presente tesis de grado centra su desarrollo en el módulo anteriormente expuesto.

[5]Sitios a los cuales se le da mantenimiento constante.

[6]Indicado por el aporte al conocimiento y no por el valor monetario

Dada la necesaria presencia de un Módulo de Categorización Automática de Imágenes Pornográficas (MCAIP) en el MCADHTML surge el siguiente problema científico ¿Cómo diseñar e implementar un categorizador automático de imágenes pornográficas?

Por lo expuesto se defiende la siguiente idea: Si se implementa un categorizador automático de imágenes pornográficas el MCADHTML aumentará su efectividad, siendo así más confiable la categorización de los documentos HTML.

La clasificación automática se refiere a un proceso que se ejecuta de forma independiente, para esto es necesario el uso de técnicas de Inteligencia Artificial (IA), por lo que se enmarcó el objeto de estudio de este trabajo en el reconocimiento de patrones y como campo de acción el reconocimiento de imágenes pornográficas, dado que es una linea de investigación bien definida dentro del objeto de estudio mencionado.

El presente trabajo se plantea como objetivo general diseñar e implementar un categorizador automático de imágenes pornográficas, desglosandoce este en los siguientes objetivos específicos:

- Aplicar los patrones de diseño que se correspondan con el problema dado.
- Aplicar un modelo de segmentación de la imagen por el color de la piel.
- Obtener los descriptores mas relevantes de las imágenes pornográficas.
- Elegir un algoritmo de entrenamiento y clasificación eficiente.

Para cumplir con los objetivos planteados se han definido las siguientes tareas investigativas:

- Estudiar trabajos similares para poner en practica la reutilización de componentes.
- Estudiar herramientas libres para acelerar el desarrollo del presente trabajo.
- Estudiar el proceso de reconocimientos de patrones en imágenes pornográficas.
- Estudiar los posibles patrones de diseño a aplicar en la solución.
- Estudiar diferentes formatos de fichero de configuración.
- Estudiar del modelo de john y rehg para la segmentación de imágenes.

- Estudiar las posibles características a extraer a la imagen.
- Estudiar las Redes Neuronales Artificiales como algoritmos de aprendizaje supervisado.

Diseño metodológico

Para el desarrollo de trabajo nos apoyamos en los siguientes métodos teóricos:

Analítico-Sintético: Se aplicó para entender el proceso de categorización de imágenes a partir del estudio del objeto del problema.

Modelación: Se uso mediante UML para reflejar la estructura, relaciones internas y características de la solución a través de diagramas.

Observación: Se uso para la selección de los descriptores de las imágenes pornográficas mediante la observación de las imágenes segmentadas, usado también para seleccionar el umbral del mapa de probabilidades del color de la piel, mediante la observación de imágenes segmentadas con diferentes umbrales.

Estructura del contenido

Para entender de una mejor forma el presente documento es necesario entender su organización y algunas reglas que se han definido para su mejor comprensión, estas son:

- Las palabras en otro idioma se escribirán en letra *cursiva*.
- Las palabras con letras MAYÚSCULA representarán acrónimos y estarán descritas en el Glosario de términos.
- Los nombres de los componentes de los diagramas presentados estarán en idioma ingles.

El documento se encuentra organizado en tres capítulos tal como se describe a continuación:

El primer capítulo esta centrado en el estado del arte relacionado con el campo de acción. En el se analiza la existencia de soluciones a nivel nacional e internacional que puedan reutilizarse. Se presenta una breve descripción del objeto de estudio. Además se identifican las tecnologías y herramientas, así como la metodología para el desarrollo del software.

En el segundo capítulo se profundiza en el problema a resolver a través de su descripción y se muestra el modelo de dominio generado. Se presenta una descripción detallada del proceso de clasificación llevado a cabo en nuestro problema específico. Se realiza una propuesta de solución y se identifican los requisitos funcionales y no funcionales que deben tenerse en cuenta. Por ultimo se hace una descripción de los casos de uso obtenidos a partir de los requisitos y se presentan los diagramas relacionados con estos.

En el tercer capítulo se presentan los artefactos generados en el flujo de análisis y diseño siendo estos los diagramas de interacción por casos de usos. Se describen los patrones de diseño utilizados en el software para mejorar su arquitectura. Además se presenta el diagrama de componentes así como un diagrama de despliegue de la aplicación.

Capítulo 1

Fundamentación teórica

1.1. Introducción

Aunque los categorizadores automáticos de imágenes pornográficas pueden existir como software autónomos[1] su uso más frecuente es formando parte, como un módulo más, de un sistema de filtrado de contenidos. Estos categorizadores de imágenes pornográficas tienen una gran importancia ya que de el depende en buena medida que sean bloqueados los sitios Web que presentan contenidos pornográficos.

1.2. Actualización

Un Filtro de Contenido es uno o mas elementos de software que operan juntos para regular (permitir o denegar) el acceso de los usuarios a determinados materiales que se encuentran en Internet. FILPACON constituye una solución de este tipo y ante la necesidad de un categorizador de imágenes pornográficas para el mismo, se precisa el estudio de soluciones similares que puedan existir.

1.2.1. Ámbito internacional

En el marco particular de los sistemas de filtrado, el uso de categorizadores de imágenes para llenar sus base de datos no está del todo extendido, de echo existen sistemas de filtrado reconocidos a escala mundial que no analizan las imágenes para dar una conclusión de una página Web analizada[3].

Otros productos si hacen un análisis tanto del texto como de las imágenes de una páginas Web para dar un criterio de esta. Estos categorizadores de textos e imágenes de los productos que lo poseen han

[1] Como es el caso de Image-filter™ perteneciente a LTU technology http://www.ltutech.com/en/technology-and-products.image-filter.html

influido en gran medida a la aceptación de los mismos ya que al utilizar técnicas de IA se reduce la dependencia de las listas de URLs logrando que el filtro en cuestión se adapte a la naturaleza dinámica de Internet. a continuación mostramos una pequeña lista de algunos sistemas de filtrado que poseen un analizador de imágenes:

- POESIA[2]
- OPTENET[3]
- PROVENTIA[4]
- Netsweeper[5]

Con la excepción de POESIA los filtros mencionados, y por ende sus categorizadores, son software privativos teniendo estos como desventajas las siguientes:

- No son productos separados del filtro.
- Son diseñados como una solución a la medida del propio filtro imposibilitando el uso de estos en otro entorno.
- Al no tener acceso al código fuente es imposible su mantenimiento.
- Alta dependencia de los propietarios.

Image-filter^{TM} es un categorizador de imágenes con una efectividad del 96 % y puede analizar de forma automática el contenido de imágenes y videos. Está diseñado para la integración con otros productos (OEM) para lograr una solución de filtrado para negocios o ISPs[4]. Este categorizadores a pesar de tener relevantes ventajas al igual que los mencionados es un software privativo.

POESIA es un proyecto diseñado para ser aplicado en el sistema operativo GNU/Linux bajo la licencia libre GPL[5]. Dicho proyecto ha supuesto la creación de un filtro de contenidos pornográficos, violentos y xenófobos basado en el filtrado tanto de contenidos textuales como de las imágenes. A pesar de las ventajas mencionadas este filtro tiene también sus desventajas:

[2] http://www.poesia-filter.org
[3] http://www.optenet.com/es/webfilter.asp?c=1
[4] http://www.virtual.com/pages/partners_iss.aspx
[5] http://www.netsweeper.com/Developers/Categorization

- En cuanto al filtrado de imágenes presenta un acercamiento pobre (43 %)[6].
- Su ultima versión disponible data del año 2006[6] lo que supone un bajo mantenimiento del software.

1.2.2. Ámbito nacional

En el país FILPACON constituye la única solución de su tipo, implicando novedad en cada uno de sus componentes. No se tienen referencias de un categorizador automático de imágenes pornográficas desarrollo por ninguna institución del país. por lo antes expresado se llega a la conclusión de que a escala nacional no existe una solución a nuestro problema.

1.3. Reconocimiento de patrones

Ya que el problema que nos ocupa está enmarcado en la IA se hace necesario un estudio sobre el tema. Hoy día la IA constituye una solución elegante para diversos problemas de la vida real con apreciables ventajas. Automatizar una tarea usando técnicas de IA hace que aumente la eficiencia de esta ya que se realiza las 24 horas del día; eliminan errores, que los humano cometen por agotamiento físico, en el proceso automatizado; generalmente con un ordenador de nuestros tiempos se pueden automatizar tareas complejas dado el avance del hardware y software; en la mayoría de las ocasiones un solo ordenador es más eficiente, a la hora de realizar la tarea específica para la cual fue concebido, que un humano.

El reconocimiento de patrones es campo dentro de la IA más cercano a nuestro problema por lo que nos enmarcaremos en este para la realización de nuestro estudio. Siguiendo la definición de Watanabe, un patrón es una entidad a la que se le puede dar un nombre y que está representada por un conjunto de propiedades medidas y las relaciones entre ellas, esto es llamado vector de características[7]. Por ejemplo, un patrón podría ser una imagen de una figura x y su vector de características serían valores numéricos relacionados con la figura x como su forma, tamaño y posición. Un sistema de reconocimiento de patrones tiene uno de los siguientes objetivos:

- Identificar el patrón como miembro de una clase ya definida[7]

[6] http://sourceforge.net/projects/poesia

[7] clasificación supervisada.

- Asignar el patrón a una clase todavía no definida[8]

El reconocimiento de patrones es ya una ciencia consolidada y cuenta con un flujo bien definido para su uso. El diseño de un sistema de reconocimiento de patrones se lleva a cabo normalmente en cuatro fases como se ilustra en la figura 1.1

Figura 1.1: Flujo para el reconocimiento de patrones.

Paso 1. Adquisición: Carga los datos que se analizarán en una variable. En ocasiones estos datos se encuentran de forma analógica y es una responsabilidad de este paso llevarlo a digital.

Paso 2. Extracción de características: Se extraen los descriptores mas relevantes del objeto analizado. Estos descriptores son seleccionados empíricamente o mediante la utilización de heurísticas.

Paso 3. Toma de decisiones o Agrupamiento: Se entrena, de forma manual[9] o automática[10], dado un conjunto de vectores provenientes del paso 2. Este paso se ejecuta solo en la primera iteración; una ves que se ejecute este, el software quedará listo para la categorización.

Paso 4. Categorización: Se le asigna una clase al vector resultante del **paso 2**. Este paso se ejecuta a partir de la segunda iteración.

Como se pudo percatar los pasos 3 y 4 se ejecutan en diferentes iteraciones. Para entrenar un software de reconocimiento de patrones se ejecutan los pasos 1, 2 y 3. Una vez entrenado el sistema se puede categorizar ejecutando los pasos 1, 2 y 4.

[8] clasificación no supervisada, agrupamiento o clustering

[9] En este coso se necesita de una colección de entrenamiento, es decir, una muestra representativa previamente clasificadas por clase, del dato específico que se desee categorizar

[10] Se refiera a la agrupación no supervisada

1.3.1. Reconocimiento de patrones en imágenes

El procesamiento de imágenes es un campo del reconocimiento de patrones por lo que tienen objetivos y características comunes. El flujo definido para el procesamiento de imágenes es similar al del reconocimiento de patrones con la excepción que incluye un nuevo paso, la segmentación, que se inserta en el segundo paso; quedando ahora el proceso de categorización como se muestra en la figura 1.2.

Figura 1.2: Flujo para el reconocimiento de imágenes.

La segmentación de la imagen es el proceso que subdivide una imagen en regiones homogenias. Cada región homogenia constituye una parte o un objeto en la escena. En otras palabras, la segmentación de la imagen es definida como la división de regiones que están conectadas, entonces cada píxel en la imagen adquiere una única etiqueta que indica a la región que pertenece. La segmentación es uno de los más importantes elementos en el reconocimiento de imágenes, principalmente porque en este paso los objetos de interés son extraidas de la imagen[8, p. 131]. Estos objetos conforman los descriptores que formarán el vector de características, el cual es pasado a los siguientes subprocesos. Existen numerosos algoritmos de segmentación mucho, de los cuales son muy eficientes pero complicados, por lo que entenderlos nos llevaría un tiempo mayor que el que tenemos concebido para la investigación. Por lo anterior planteado hemos decidido adoptar uno de los algoritmos de segmentación más simples, la segmentación por umbral[11].

1.3.2. Redes Neuronales Artificiales

Un paso fundamental en un categorización automática es el entrenamiento por lo que se debe seleccionar el algoritmo adecuado para esto. Teniendo en cuenta que vamos a identificar un patrón como miembro de una clase ya definida estamos en presencia de clasificación supervisada. Para este tipo de

[11] Para buscar este método en la bibliografía anglosajona se debe de usar el termino **thresholding segmentation**

clasificación se estudiaron los algoritmos k-NN y RNA de los cuales se seleccionó el RNA por ser un algoritmo, para nuestro caso en específico, más eficiente que el k-NN, y fácil de aplicar.

Las RNA tienen un extensivo uso en clasificación de objetos. Estas redes son entrenadas, las cuales son usadas para clasificar objetos en una escena. Un gran número de neuronas interconectadas, que se ejecutan en paralelo para mejorar la tarea de aprendizaje. Basado en el tipo de aprendizaje, estas redes pueden ser supervisadas o no supervisadas aunque la que se utilizará será una red supervisada, concretamente un Perceptrón Multicapa o MLP.

1.3.3. Aplicaciones del procesamiento de imágenes

El procesamiento de imágenes tiene un gran número de aplicaciones en muchas áreas de las actividades humana, a continuación se mencionan alguna de ellas. Es valido aclarar que las áreas de aplicación del reconocimiento de imágenes presentadas representan una pequeña parte del total de estas.

Seguridad: Mediante el reconocimiento de patrones biométricos como son el rostro, las hullas dactilares, el iris del ojo entro otros.

Medicina: Mediante el análisis de imágenes, la cual pueden ser imágenes de resonancia magnética, para el diagnostico de enfermedades.

Industria: Mediante la detección de fallos, a través de una imagen, en componentes industriales como por ejemplo, la detección de fallos en filamentos de lamparas incandescentes.

Filtrado: Mediante la categorización de imágenes que se desean filtrar como por ejemplo, las imágenes pornográficas, símbolos, rostros, objetos entro otros.

1.4. Tecnologías a utilizar

El uso apropiado de herramientas puede mejorar la efectividad y eficiencia con la que se desarrollan los software; al mismo tiempo, su uso beneficia la calidad del mismo. La automatización puede aumentar los beneficios obtenidos con el uso de las herramientas al disminuir el tiempo necesario para terminar un proyecto, al evitar el tedio asociado con ciertas tareas y al seguir procedimientos consistentes.

1.4.1. Lenguaje de programación

Un lenguaje de programación es un conjunto limitado de palabras y símbolos que representan procedimientos, cálculos, decisiones y otras operaciones que pueden ejecutar una computadora. Los mismos se pueden dividir en dos grupos fundamentales: Descriptivo (Declarativo) ó Prescriptivo (Imperativo)[12]. Existen otras agrupaciones para los lenguajes de programación y una de las más comunes es su agrupación por lenguajes de alto y bajo nivel.

Hoy día existen numerosos lenguajes de programación, cada uno de ellos fueron creado para un fin determinado, por los que algunos son mejores que otros a la hora de realizar una tarea específica. Elegir un lenguaje de programación para el desarrollo de una aplicación no es tarea sencilla ya que hay que tener en cuenta muchos aspectos para saber a priori si el seleccionado es el adecuado para resolver el problema dado. Una de las posibles causas del fracaso de un proyecto es una mala selección del lenguaje de programación que se usará.

El presente trabajo propone para su desarrollo al lenguaje de programación C++ elegido fundamentamente por ser uno de los más usados a escala mundial[9], su abundante documentación, cuenta con muchas bibliotecas para solucionar diversos problemas, lo que agiliza el desarrollo, y por la experiencia de los desarrolladores en este lenguaje, lo que permite un desarrollo más fluido.

Breve historia de C++

El lenguaje C++ está basado en el lenguaje de programación C, el cual a su vez está basado en dos lenguajes muy primitivos (BCPL y B). Sus fundadores fueron Martin Richards[13] (1976) y Ken Thompson[14] (1970) respectivamente. Dos años mas tarde de la creación del lenguaje B Dennis Ritchie implementó el diseño de BCPL, B y creó C; que se dio a conocer por ser el lenguaje de programación de desarrollo de UNIX.

A principio de la década del 80, Bjarne Stroustrup[15] empezó a desarrollar C++, que recibiría formalmente su nombre a finales de 1983. En octubre de 1985, apareció la primera divulgación comercial del

[12] los cuales a su vez se dividen en numerosos subgrupos

[13] http://www.cl.cam.ac.uk/~mr10

[14] http://www.cs.bell-labs.com/who/ken

[15] http://www.research.att.com/~bs

lenguaje y la primera edición del libro *The C++ Programming Language*[16].

Desde ese momento, C y C++ se han usado para la creación de sistemas operativos, por lo que su popularidad ha ido creciendo a lo largo de los años, y actualmente es uno de los lenguajes de programación más usados por los programadores.

Características del lenguaje

C++ cuenta con características importantes en las cuales nos apoyamos para su selección como lenguaje de programación para el desarrollo de la aplicación, estas características se describen a continuación:

Alto nivel: Este tipo de lenguaje permite al programador abstraerse de la arquitectura y funcionamiento del hardware.

Orientado a objetos: La posibilidad de orientar la programación a objetos permite al programador diseñar aplicaciones desde un punto de vista más cercano a la vida real. Además, permite la reutilización del código de una manera más lógica y productiva.

Portabilidad: Un código escrito en C++ puede ser compilado en muchos sistemas operativos[17].

Brevedad: El código escrito en C++ es muy corto en comparación con otros lenguajes, sobretodo porque en este lenguaje es preferible el uso de caracteres especiales que las “palabras clave”.

programación modular: Un cuerpo de aplicación en C++ puede estar hecho con varios ficheros de código fuente que son compilados por separado y después unidos. Además, esta característica permite unir código en C++ con código producido en otros lenguajes de programación como Ensamblador o el propio C.

Velocidad: El código resultante de una compilación en C++ es muy eficiente, gracias a su capacidad de actuar como lenguaje de alto y bajo nivel y a la reducida medida del lenguaje.

[16] Escrito por el propio Bjarne Stroustrup

[17] Para esto es necesario usar el estándar ANSI para C++. En ocasiones deben hacerse pequeños cambios.

1.4.2. Bibliotecas disponibles

Dado que los softwares van ganando en complejidad cada día, se hace necesaria la busqueda y utilización de herramientas que agilicen el desarrollo de software, como son las bibliotecas, paquetes de clases, frameworks, módulos, etc. C++ es un lenguaje con una gran historia por lo que cuenta con muchas bibliotecas para la solución de disimiles problemas. Enmarcandonos en el problema que nos ocupa se necesita de una biblioteca capaz de manejar imágenes. Mediante el estudio realizado se descubrió la existencia de una biblioteca con excelentes características para el procesamiento de imágenes llamada LTI-Lib cuyas características se exponen a continuación:

- Licenciada bajo la GPL
- Desarrollada con el estándar ANSI para C++
- Implementa varios algoritmos de segmentación (thresholding, mean shift, SOFM)
- Implementa varios algoritmos de entrenamiento y clasificación (MLP, SOFM, k-NN)

Las características anteriores son suficientes para resolver el problema presentado en este trabajo por lo que adoptamos la librería LTI-Lib como base para el desarrollo de un clasificador de imágenes pornográficas. Otra biblioteca utilizada es la STL la cual brinda muchas funcionalidades de interés como es el trabajo con cadenas (string), excepciones y manejo de la entrada y salida estándar.

1.4.3. IDE de desarrollo

Las ventajas principales de un IDE radica en la automatización de algunas tareas asociadas al desarrollo de aplicaciones y al soporte para algunas funcionalidades que facilitan el desarrollo de software disminuyendo así el tiempo de construcción. La selección un *Integrate Development Enviroment* (IDE) para el desarrollo de aplicaciones en C++ en el sistema operativo GNU/Linux se hace hasta cierto punto difícil, ya que ninguno de los IDEs existentes completa código lo cual es una de las características que más agiliza el desarrollo de software al no tener que memorizar y teclear todas las funciones existentes en otra clase. Finalmente encontramos uno que si hace esta tarea y es el caso de Eclipse con un pluging llamado CDT el cual tiene las siguientes características:

- Editor de C/C++ (Funcionalidades básicas, resaltamiento de sintaxi, completamiento de código, etc)

- Depurador C/C++ (Usando GDB)
- Lanzador C/C++ (Lanzadores y aplicaciones externas)
- Parser[18]
- Herramienta de busqueda
- Proveedor de asistencia de contenido
- Generador de Makefile[19]

Adicionalmente a estas características mencionadas, pertenecientes al pluging CDT; se tienen otras ventajas del Eclipse, como IDE de desarrollo, las cuales meresen ser mencionadas:

- Posee una excelente integración con SVN[20]
- Posee un potente administrador de proyecto
- Es multiplataforma
- Altamente configurable

La única desventaja hasta el momento es su gran consumo de recurso. Para su uso se recomienda un ordenador con al menos 512 MB de RAM con un procesador a una velocidad de 2.4 GHz o superior.

1.4.4. Herramienta CASE

El solo hecho de imaginar la creación de los diagramas de un ciclo de vida de RUP a lápiz y papel demuestra la necesidad e importancia de estas herramientas. Se seleccionó la herramienta CASE BoUML teniendo en cuenta sus características y nuestras necesidades.

- Es software libre, licenciado bajo la GPL
- Es muy ligero, necesita escasos recursos computacionales

[18] Programa analizador sintáctico

[19] Un fichero de configuración usado por el programa **make** definiendo la ubicación del código fuente, como deberá ser compilado y vinculado para crear un programa ejecutable.

[20] Control de versiones SubVertion

- Genera el diagrama de clases de diseño a partir de un código fuente (inversión de código)
- Genera el código fuente a partir del diagrama de clases del diseño (generación de código)
- Genera la documentación del proyecto (en HTML)
- Importa proyectos del Rational Rose
- Posibilita la realización de los siguientes diagramas:
  - Caso de uso
  - Objeto
  - Estado
  - Actividades
  - Colaboración
  - Secuencia
  - Clase
  - Componente
  - Despliegue

1.4.5. Otras herramientas

Es justo mencionar algunas herramientas auxiliares utilizadas para confección del presente documento que, aunque no formanron parte del desarrollo del software, si contribulleron a disminuir el tiempo y aumentar la calidad del trabajo; ellas son:

LaTeX: Es un sistema tipográfico para generar documentos científicos de alta calidad, lo cual permitió centrarnos en el documento dejándole a este la tarea de generar un documento que cumpliera con la calidad requerida para una tesis de grado.

Kile: Es un IDE para LaTeX el cual posibilitó un desarrollo fluido del documento.

Dia: Es un software de dibujo utilizar para generar las figuras del presente documento.

Gimp Es un software de edición de imágenes. Usado en el desarrollo del documento para editar las imágenes generadas por **dia** y mejorar su estética. Usando en el desarrollo del software para observar el histograma de una imagen, selección de umbral, cortar una región de una imagen entro otras tareas.

1.5. Metodología de desarrollo

Una metodología de desarrollo de software es un proceso, el cual define qué, cuándo, y cómo alcanzar un determinado objetivo. Este debería ser capaz de servir como guía para todos los participantes

(clientes, usuarios, desarrolladores y directores ejecutivos), evolucionar durante muchos años permitiendo limitar su alcance en un momento del tiempo dado a las realidades de la tecnología, herramientas, personas y patrones de organización.

Se puede decir que, una metodología de desarrollo de software es un proceso que guía a los desarrolladores a los que les brinda métodos y herramientas, proporcionándoles una ayuda muy importante e indispensable para que el producto final posea las funcionalidades requeridas por el cliente y de que cumpla con las necesidades del mismo y del usuario final, es una secuencia de actividades organizadas y bien pensadas que transforman los requisitos del cliente en el producto final.

1.5.1. RUP

Teniendo en cuenta la necesidad e importancia de utilizar una metodología, en el caso particular del MCAIP perteneciente al MCADHTML se decidió utilizar *Rational Unified Process* (RUP). La decisión final quedo sustentada por los siguiente elementos:

- Es una de las metodologías más usadas y mayormente probadas a nivel internacional para el desarrollo de software.
- Mantener un proceso homogéneo entre el desarrollo del MCAIP y FILPACON, ya que este usa RUP para la gestión de sus procesos.
- La experiencia del equipo de desarrollo.

Características de RUP [10]

- **Dirigido por Casos de Uso**: Tiene a los Casos de uso como el hilo conductor que orienta las actividades de desarrollo. Se centra en la funcionalidad que el sistema debe poseer para satisfacer las necesidades de un usuario (persona, sistema externo, dispositivo) que interactúa con él.
- **Centrado en la arquitectura**: Propone arquitectura de forma similar a la de un edificio. Es necesario tener varios planos con diferentes aspectos, para tener una imagen completa del edificio antes que comience su construcción, aquí entra a jugar el término Arquitectura de Software, que abarca las diferentes vistas que se mencionaron anteriormente.

- **Iterativo e incremental**: Propone la descomposición de proyectos grandes en proyectos más pequeños o subproyectos, cada uno de estos subproyectos es una iteración, y cada iteración debe estar controlada y tratar un determinado grupo de casos de uso.

RUP provee un acercamiento a disciplinas para asignar tareas y responsabilidades en un desarrollo organizado. Su objetivo es asegurar la producción de software de alta calidad que conozca la necesidad de sus clientes dentro de una predecible espacio de tiempo y presupuesto. La figura 1.3 ilustra la arquitectura de RUP, la cual tiene dos dimensiones:

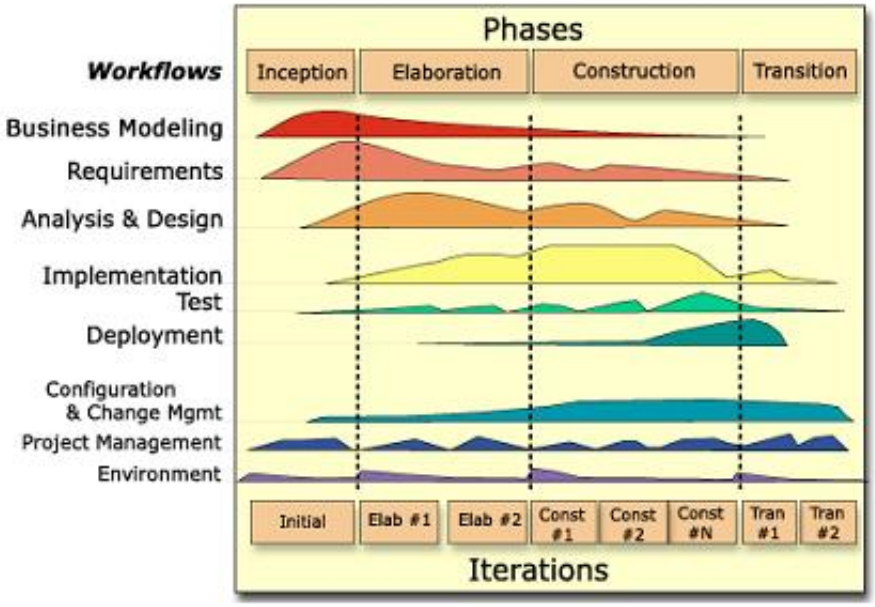

Figura 1.3: Arquitectura de RUP

Faces de desarrollo

1. **Inicio:** Determinar la visión del proyecto.
2. **Elaboración:** Determinar la arquitectura óptima.
3. **Construcción:** Llegar a obtener la capacidad operacional inicial.
4. **Transmisión:** Llegar a obtener el una primera versión del proyecto (release).

Cada una de estas etapas es desarrollada mediante el ciclo de iteraciones, la cual consiste en reproducir el ciclo de vida en cascada a menor escala. Los Objetivos de una iteración se establecen en función de la evaluación de las iteraciones precedentes. Vale mencionar que el ciclo de vida que se desarrolla por cada iteración, es llevada bajo dos disciplinas:

Disciplina de Desarrollo

Ingeniería de Negocios: Entendiendo las necesidades del negocio.

Requerimientos: Trasladando las necesidades del negocio a un sistema automatizado.

Análisis y Diseño: Trasladando los requerimientos dentro de la arquitectura de software.

Implementación: Creando software que se ajuste a la arquitectura y que tenga el comportamiento deseado.

Pruebas: Asegurándose que el comportamiento requerido es el correcto y que todo los solicitado esta presente.

Disciplina de Admitidas

Configuración y administración del cambio: Guardando todas las versiones del proyecto.

Administrando el proyecto: Administrando horarios y recursos.

Ambiente: Administrando el ambiente de desarrollo.

Distribución: Hacer todo lo necesario para la salida del proyecto

Elementos del RUP

Actividades: Procesos que se llegan a determinar en cada iteración.

Trabajadores: Las personas o gentes involucrados en cada proceso.

Artefactos: Un artefacto puede ser un documento, un modelo, o un elemento de modelo.

Una particularidad de esta metodología es que, en cada ciclo de iteración, se hace exigente el uso de artefactos, siendo por este motivo, una de las metodologías más importantes para alcanzar un grado de certificación en el desarrollo del software.

1.6. Conclusiones

Luego de analizar los software capaces de categorizar las imágenes pornográficas en los ámbitos internacional y nacional se evidencia la necesidad de diseñar e implementar uno propio ya que los existentes, en el ámbito internacional, no cumplen no cumplen con las características requeridas por FILPACON y en el ámbito nacional no existen. Mediante el estudio de las técnicas de IA, herramientas, tecnologías y metodologui a utilizar, se han sentado las bases para el desarrollo de una aplicación capas de clasificar de forma automática las imágenes en pornográficas y no pornográficas. A continuación presentamos algunos aspectos importante definidos en el presente Capítulo.

1.6.1. Tecnologías

Se adoptaron las siguientes tecnologías para el desarrollo del trabajo.

- como lenguaje de programación: C++
- como biblioteca: LTI-Lib
- como IDE de desarrollo: Eclipse
- como herramienta CASE: BoUML

1.6.2. Inteligencia Artificial

Se estudiaron los algoritmo que componen los diferentes subprocesos para la categorización de imágenes, seleccionando los siguientes, en los cuales se basará nuestro trabajo.

- como algoritmo de segmentación: segmentación por umbral
- como algoritmo de entrenamiento: MLP

Capítulo 2

Características del sistema

2.1. Introducción

Para guiar el desarrollo de MCAIP hacia el sistema correcto es necesario comprender su contexto e identificar las condiciones o capacidades que debe cumplir. En este capítulo, para definir las características del sistema: se realiza el modelado del dominio con el objetivo de comprender su contexto; se hace su propuesta, describiendo cómo debe funcionar y destacando sus características distintivas; se especifican sus requisitos funcionales y no funcionales, además se elabora el Modelo de Casos de Uso, definiendo sus actores, casos de uso y las relaciones entre ellos.

2.2. Descripción del problema

Actualmente, para crear y actualizar la BDUC de FILPACON, se han creado programas que compendian las listas de URLs categorizadas por terceras partes. Para evitar esta inconveniente dependencia, al Grupo **API!** le asignaron el desarrollo de MCADHTML.

MCADHTML incluirá, entre otros, el MCAIP y el Módulo Controlador; este último se encargará de gestionar los procesos que en MCADHTML ocurran. La comunicación entre ambos módulos será bidireccional. El Módulo Controlador le enviará al MCAIP una lista[1] de imágenes a categorizar, y este le devolverá un lista de categorías correspondientes con la lista de imágenes enviadas.

Para que el Módulo MCAIP clasifique las imágenes enviadas por el Módulo Controlador necesita de estar dotado de “inteligencia”, la cual es adquirida por el sistema mediante el empleo de técnicas de

[1] Es una ruta a una carpeta la cual contiene las imágenes a categorizar. ej. `/usr/share/pixmaps`

IA. En el Capítulo anterior se describió de forma general el proceso de de aprendisaje y clasificación sentando las bases para su profundización en el presente capítulo.

2.3. Modelo de dominio

Por la relativa simplicidad del entorno donde está enmarcado el sistema y por el conocimiento que poseen los desarrolladores acerca del funcionamiento del mismo, no es necesario profundizar a través de un modelo de negocio. Es suficiente con un Modelo de Dominio para capturar los principales elementos envueltos en la problemática que la aplicación resuelve. La figura 2.1 muestra el Modelo de Dominio realizado teniendo en cuenta lo plantedo en la descripción del problema.

El Modelo de Dominio (o Modelo Conceptual) es una representación visual de los conceptos u objetos del mundo real significativos para un problema o área de interés. Representa clases conceptuales del dominio del problema, conceptos del mundo real y no de los componentes de software[11].

Identificar muchos objetos o conceptos forma parte de una investigación del problema. El lenguje UML contiene la notación en diagramas de estructura estática que explican gráficamente los Modelos de Dominio. El paso esencial de un análisis orientado a objetos es descomponer el problema en conseptos individuales; ya que es una representación de conseptos en un dominio del problema lo ilustramos con un grupo de diagramas de estructura estática donde no se define ningura operación[12, p. 87]. Puede mostrarnos:

- conceptos
- asocicicones en conceptos
- atributos de conceptos

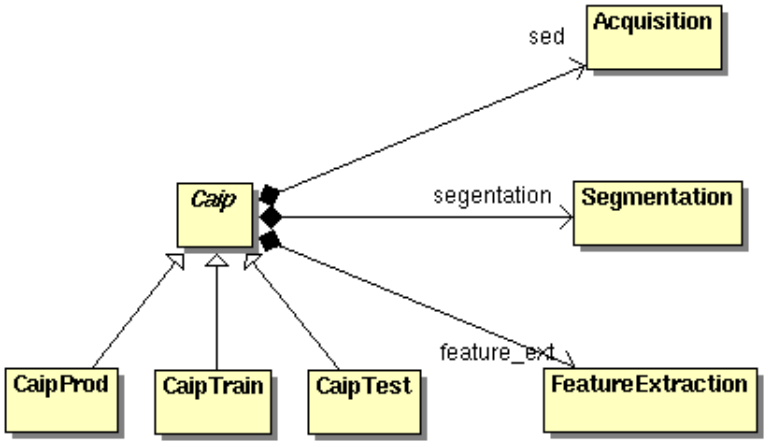

Figura 2.1: Modelo de Dominio del problema.

2.4. Solución propuesta

Se propone un Módulo de Categorización Automática de Imágenes Pornográficas basado en técnicas de Inteligencia Artificial, el cual tendrá dos modos de funcionamiento: conectado y no conectado. En el modo conectado estará interactuando con el Módulo Controlador dentro de MCADHTML; mientras que en modo no conectado un usuario común podrá ajustarlo, efectuando entrenamientos y pruebas hasta lograr la configuración deseada. Esta aplicación no contará con una IGU dado que, en primer lugar, cuando trabaja en modo conectado el Módulo Controlador se comunican con la aplicación mediante una mediante una tubería[2] lo cual no necesita una interfaz gráfica; en segundo lugar, cuanto trabaja en modo no conectado su comportamiento es gestionado mediante un mediante un fichero de configuración el cual es simple y trivial[3].

El fichero de configuración mencionado permitirá configurar el software para satisfacer las necesidades de los usuarios de forma sencilla, además la ubicación de este fichero no influirá en absoluto ya que su ruta es pasada por la entrada estándar al categorizador cuando se inicia este, así se podrá tener varios

[2]Es conocido como *pipe* y su función es direccionar la salida de un programa a la entrada de otro, esto se hace mediante la salida y entrada estándar

[3]Básicamente posee 13 parametros.

ficheros de configuración con diferentes comportamientos. La propuesta de arquitectura de este fichero se muestra en el Anexo D

El sistema debe de guardar el modelo de categorización creado durante el entrenamiento y cargarlo antes de iniciar la categorización. En sentido general, el Módulo MCAIP debe proporcionar un adecuado margen de error para poder aplicarse en el MCADHTML ya que si este posee en alto margen de error las clasificaciones de las URLs serán erróneas, la Base de Datos de URLs Categorizadas perdería valides, FILPACON permitiría el acceso a contenidos inadecuados lo que lleva al fracaso del proyecto.

Aunque este trabajo esta dirigido a implementar un sistema para el MCADHTML perteneciente a FILPACON, este sistema deberá poseer una autonomía total del FILPACON permitiendo así que, este pueda ser reutilizado por otros usuarios que deseen esta funcionalidad en sus sistemas.

2.5. Categorización de imágenes

Como se describió, de forma general, en el Capítulo 1.3 la categorización de imágenes requiere cinco pasos fundamentales los cuales se describen a continuación, enfocandonce en el caso específico de este trabajo que es el reconocimiento de imágenes pornográficas.

2.5.1. Adquisición

La adquisición es el proceso de obtención de la imagen. Supondría un problema si estas se encontrara en forma analógica, en nuestro caso esto no es mayor problema ya que las imágenes provienen de Internet y estas se encuentran en formato digital. Existe una larga lista de formatos de imágenes lo que constituye un problema pues estos distintos formatos se leen de forma diferente y se debe tener una biblioteca que admita al menos la lectura para cada uno de ellos. Hasta el momento la LTI-Lib, que es la biblioteca utilizada, solo admite los formatos JPEG y PNG. Dichos formatos son unos de los más usados en Internet[13] por lo que no se precisa, por el momento, de una bibliotecas para leer otro formato de imagen.

2.5.2. Segmentación

El objetivo principal de este paso es separa las partes principales de una imagen. La segmentación autónoma[4] es uno de los procesos más difíciles en el procesamiento de imagen. Por una parte una segmentación adecuada nos facilitará mucho la solución del problema y una segmentación errónea nos conducirá a fallos. Una completa segmentación de una imagen R comprende la identificación de un grupo finito de regiones $\langle R_1, R_2, R_3, \cdots, R_N \rangle$ tal que:

1. $R = R_1 \cup R_2 \cup \cdots R_N$
2. $R_i \cap R_j = \Phi$, $\forall i \neq j$
3. $P(R_i) =$ Verdadero, $\forall i$
4. $P(R_i \cup R_j) =$ Falso, $i \neq j$

La segmentación en nuestro sistema es realizada a partir del color de la piel humana dado que las imágenes pornográficas muestran generalmente personas desnudas. Estas imágenes muestran mucha piel y debido a esto el color de la piel es una característica básica usada para la categorización de tales imágenes. Una desventaja que puede traer escoger está característica como primaria es que no se podrá trabajar con imágenes en niveles de grises, sin embargo las imágenes del dominio que nos ocupa raramente están en blanco y negro.

Modelado del color de la piel

Existen numerosos trabajos relacionados con el modelado del color de la piel, que abarcan definición implícita, métodos paramétricos y no paramétricos. También se usan diferentes espacios de color, siendo los más usados el RGB, RGB normalizado y HSV[14]. El objetivo de esta tarea es etiquetar cada píxel de la imagen con dos valores posibles: 1 si el píxel posee color de la piel, 0 si el píxel no posee color de la piel. Como resultado se obtiene una imagen binaria o donde se distinguen claramente la regiones correspondientes a la piel humana. Empezamos nuestro trabajo usando un modelo explícito del color de la piel en el espacio RGB, que clasificaba un píxel según el siguiente criterio:

[4] Sin necesidad de la intervención de un humano

pixel$[R, G, B]$ tiene color de la piel si:

$$R > 95 \text{ y } G > 40 \text{ y } B > 20 \text{ y}$$

$$max\{R, G, B\} - min\{R, G, B\} > 15$$

$$\text{y } \mid R - G \mid > 15 \text{ y } R > G \text{ y } R > B$$

Sin embargo el método propuesto por Jones y Regh ofrece resultados mucho mejores; por lo cual terminamos escogiéndolo[15]. Este consiste en un modelo no paramétrico del color de la piel basado en un clasificador bayesiano. Como resultado de la aplicación de este método a una imagen se obtiene un mapa de probabilidad de la piel que no es más que una imagen en escala de grises donde cada píxel posee una valor entre 0 y 1 que indica su grado de pertenencia a la clase piel.

Para realizar la segmentación se define un umbral, entre cero y uno, y todo píxel cuyo valor sea igual o mayor que dicho umbral es etiquetado como piel y como no piel en coso contrario dando como resultado la imagen binaria descrita anteriormente. El umbral que hemos escogido es 0.7. Un umbral más cercano a 0 daría como resultado muchas zonas etiquetatas con color de la piel sin serlas y uno más alto produciendo el efecto contrario.

2.5.3. Extracción de características

La extracción de característica juega un papel importante en todo sistema de reconocimiento de patrones pues nos permite obtener descriptores de los objetos para su clasificación. Las características han de ser consistentes, o sea, no se ha de tomar como características aquellas que no aporten información o que no permitan realizar una discriminación entres las diferentes clases involucradas en nuestra aplicación. De este modo no cumple objetivo alguno extraer características como el formato de la imagen, siendo relevantes sin embargo aquellas que se relacionan con el color y la forma.

Este proceso se ha dividido en dos partes, en la primera extraemos un grupo de descriptores de todas las regiones marcadas como piel, y en la segunda solo se extraen de las regiones más grandes marcadas como piel. A su vez estas característica se han dividido en dos grupos, las relacionadas con el color y las que brindan información acerca de la geometría de la mayor región.

Características relacionadas con el color:

- Cantidad de píxeles detectados con el color de la piel en la imagen
- Cociente entre la cantidad de píxeles con color de la piel y el total de píxeles de la imagen
- Cantidad de regiones con color de la piel
- Cantidad de píxeles detectados con el color de la piel en la región más grande
- Cociente entre la cantidad de píxeles con color de la piel de la región más grande y el total de píxeles con color de la piel de la imagen

Características relacionadas con la forma de la mayor región:

- Perímetro
- Relación existente entre el área y su perímetro[5]

Las característica mencionadas se seleccionaron de forma empírica, a partir de la observación de una serie de imágenes segmentadas. De esta forma queda conformado el vector característico de una imagen pornográfica.

2.5.4. Perceptrón Multicapa

Aunque existen muchos modelos y representaciones de Redes Neuronales Artificiales (RNA), cada una de ellas tienen cuatro atributos $< N_c, W, \sigma, \delta >$, donde N_c es un número finito de neuronas fuertemente interconectadas con las salidas $n_1, n_2, n_3, \cdots, N_k$; W denota un número finito de pesos los cuales representan la fuerza w_{ij} de la interconexión entre las neuronas n_i y n_j; σ es la regla de propagación que muestra como las señales de entradas a la neurona n se propagan a través de ella. Un regla típica de propagación puede ser $\sigma(i) = \sum n_i w_{ij}$ y δ es una función de activación la cual es usualmente una función no lineal como la función sigmoid[6].

El modelo más popular de redes neuronales es el MLP, el cual es una extensión de un perceptrón de una sola capa propuesto por Rosenblatt. MLP, en general, son redes retroalimentadas, teniendo distintas entradas, salida y una capa oculta. La arquitectura de la red neuronal MLP con error de propagación hacia atrás es mostrada en el Anexo C.

[5] Esta relación se nombre en la bibliografía anglosajona como compactness.

[6] http://mathworld.wolfram.com/SigmoidFunction.html

En un problema de M-clases donde los patrones son N-dimensional, la capa de entrada consta de N neuronas y la capa de salida costa de M neuronas. Puede haber una o más capas ocultas. Se considera el caso de una solo capa oculta, la cual es extensiva a cualquier número de estas. Las capas ocultas constan de p neuronas. La salida de cada neurona en la capa de entrada es conectada a las neuronas de la capa oculta. Ningún calculo es ejecutado en las neuronas de la capa de entrada. Las neuronas de la capa oculta agrupa las entradas, pasando estas por la función no lineal sigmoid y hace múltiples conexiones con las neuronas de la capa de salida[8, p.172].

Muestro problema consta de dos clases, pornográfica y no pornográfica, y de siete características. Utilizando la anterior planteado podemos concluir que nuestra red neuronal tendrá siete neuronas en su capa de entrada, una capa oculta de cuatro neuronas y en su capa de salida tendrá dos neuronas. La arquitectura de esta red es mostrada en la figura 2.2

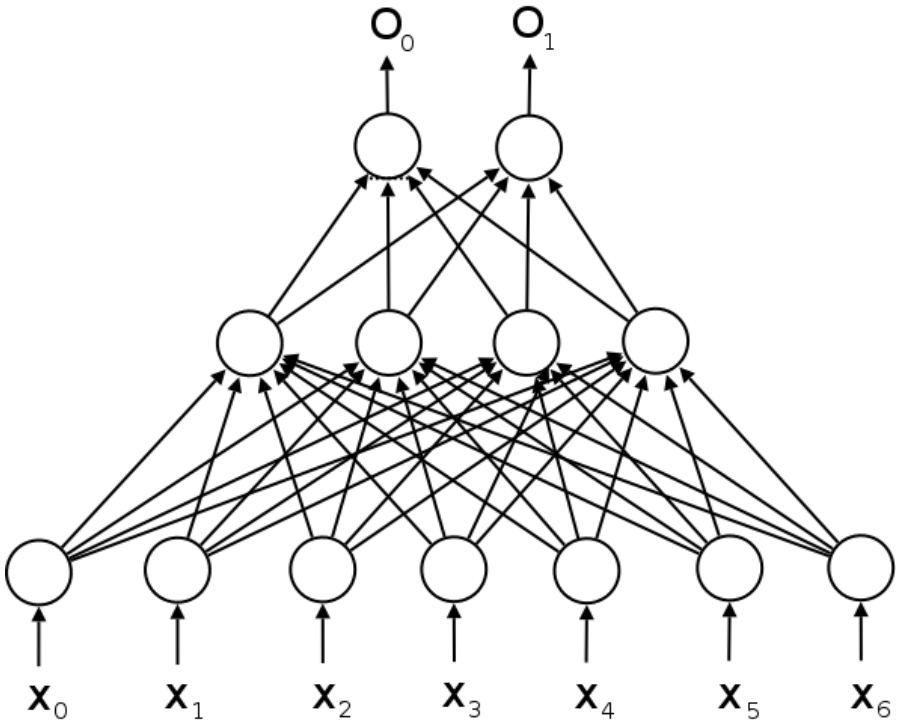

Figura 2.2: Arquitectura de la Red Neuronal utilizada para la solución del problema.

2.5.5. Categorización

Una vez entrenado el sistema se puede categorizar. Este es el último paso del proceso de categorización el cual tiene como objetivo asignar la imagen entrada a una de las clases definidas, pornográfica o no pornográfica. Para realizar la categorización se ejecutan los dos primeros pasos obteniendo el vector característico de la imagen, este vector es pasado a la Red Neuronal, en la modalidad de categorización, la cual dará como salida; mediante la activación de una de sus dos neuronas de salida, la categoría a la que pertenece la imagen. De esta forma a cada imagen se le asignará una etiqueta correspondiente a la categoría que pertenece.

2.6. Requisitos funcionales

R1. Entrenar el categorizador

R2. Categorizar lista de imágenes

R3. Probar el categorizador

2.7. Requisitos no funcionales

Los requisitos no funcionales (atributos de calidad) aseguran que se disponga de un sistema manejable y gestionable que ofrezca la funcionalidad requerida de manera fiable, ininterrumpida o con el tiempo mínimo de interrupción, incluso ante situaciones inusuales.

2.7.1. Relacionados con la calidad

El estándar *ISO/IEC 9126* define un modelo de calidad de software en el que se presentan seis características de calidad que se dividen a su vez en un conjunto de subcaracterísticas. Como uno de los objetivos de esta normativa es ayudar a identificar los requisitos de un software, entonces los elementos de calidad que ella propone se toman como requisitos no funcionales a cumplir por el sistema, siempre y cuando se mantengan las condiciones especificadas para su uso, ellos son:

1. Debe poseer una funcionabilidad adecuada, o sea, satisfacer los requisitos funcionales declarados e implícitos.

2. Debe contar con una confiabilidad adecuada de modo que mantenga el mismo nivel de ejecución a lo largo del tiempo.

3. Debe tener buena usabilidad, de modo que el esfuerzo para usarlo sea mínimo, además debe ser intuitivo para los usuarios.

4. La eficiencia debe ser alta. Además el consumo de recursos debe ser mínimo.

5. La mantenibilidad debe ser alta, de modo que pueda adaptarse a condiciones cambiantes del entorno en que se ejecute y las modificaciones puedan hacerse fácilmente.

6. Debe contar con la portabilidad necesaria para poder ser transferido de un ambiente a otro, reemplazado por nuevas versiones y además fácil de instalar y adaptar.

2.7.2. Relacionados con el entorno o ambiente

Describen como el sistema estará relacionado con su entorno y las requisitos mínimos para que pueda ser usado.

- Para ejecutar el software se necesita la biblioteca libstdc++5 | libstdc++6 | lib64stdc++6.
- Para compilar el software se necesita la biblioteca libstdc++5 y libstdc++5-dev | libstdc++6 y libstdc++6-dev | lib64stdc++6 y lib64stdc++6-dev.

2.8. Definición de los casos de uso

Los Casos de Uso son la base para la implementación de las fases y disciplinas del RUP. Un Caso de Uso es una secuencia de pasos a seguir para la realización de un fin o propósito, y se relaciona directamente con los requerimientos, ya que un Caso de Uso es la secuencia de pasos que conlleva la realización e implementación de un requerimiento planteado.

2.8.1. Definición de los actores

Actores	**Descripción**
Usuario	Usuario común el cual puede ejecutar cualquier tarea del categorizador.
Controlador	Controlador del MCADHTML y solo puede ejecutar la tarea de categorización de una lista de imágenes

Cuadro 2.1: Definición de los actores

2.8.2. Listado de casos de uso

Caso de Uso:	Entrenar el categorizador
Actores:	Usuario
Resumen:	El caso de uso se inicia cuando el actor cambia al categorizador al modo *train* en el fichero de configuración e inicia el sistema
Referencia:	R1

Cuadro 2.2: CU Entrenar el categorizadas

Caso de Uso:	Categorizar lista de imágenes
Actores:	Controlador
Resumen:	El caso de uso se inicia cuando el actor cambia al categorizador al modo *prod* en el fichero de configuración e inicia el sistema
Referencia:	R2

Cuadro 2.3: CU Categorizar lista de imágenes

Caso de Uso:	Probar el categorizador
Actores:	Usuario
Resumen:	El caso de uso se inicia cuando el actor cambia al categorizador al modo *test* en el fichero de configuración e inicia el sistema
Referencia:	R3

Cuadro 2.4: CU Probar el categorizador

2.8.3. Diagrama de casos de uso

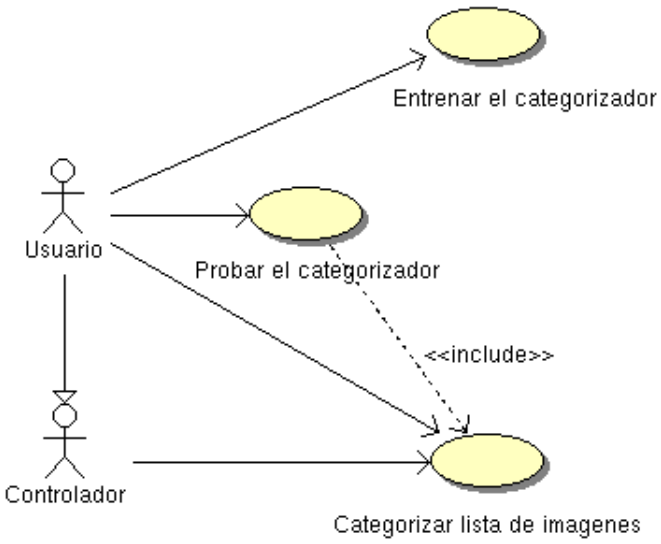

Figura 2.3: Diagrama de caso de uso del sistema.

2.9. Casos de uso expandidos

Caso de Uso:	Entrenar el categorizador
Actores:	Usuario
Resumen:	El Caso de Uso se inicia cuando el usuario decide entrenar el categorizador.
Referencia:	R1
Flujo Normal de Eventos	
Acción del Actor	**Respuesta del Sistema**
Continúa en la próxima página	

1. Abre el fichero de configuración 2. Pone al categorizador en modo de entrenamiento (*mode = train*) 3. Llena las variables referente a: 3.1. Sección de [*train*] 3.2. Sección de [*segmentation*] 3.3. Sección de [*classifier_mlp*] 4. Guarda los cambios realizados. 5. Invoca al clasificador y le pasa por la entrada estándar la ruta del fichero de configuración	6. Si se encuentran campos sin llenar o no existen las rutas a los ficheros especificados se lanza en error, a través de la salida estándar, informándole al usuario las causas del error y pidiendo volver al paso 1. 7. Segmenta las imágenes 7.1. Si es activada la variable *save_ima_mask* es guardada la imagen en escala de grises 7.2. Si es activada la variable *save_binary_img* es guardada la imagen binaria 8. Extracción sus características 9. Entrena la Red Neuronal MLP 10. Guarda el modelo de categorización (entrenamiento) en un fichero 11. Envía un mensaje, por la salida estándar, asegurando que el proceso se realizó de forma satisfactoria
Prioridad:	Crítico

Cuadro 2.5: Descripción del CU Entrenar el categorizador

Caso de Uso:	Categorizar lista de imágenes
Continúa en la próxima página	

Actores:	Controlador	
Resumen:	El Caso de Uso se inicia cuando el controlador decide categorizar una lista de imágenes.	
Referencia:	R2	
Precondiciones:	El sistema debe de estar entrenado	
Flujo Normal de Eventos		
Acción del Actor		**Respuesta del Sistema**
Continúa en la próxima página		

1. Abre el fichero de configuración 2. Pone al categorizador en modo de categorizar (*mode* = *prod*) 3. Llena las variables referente a: 3.1. Sección de [*prod*] 3.2. Sección de [*segmentation*] 3.3. Sección de [*classifier_mlp*] 4. Guarda los cambios realizados. 5. Invoca al clasificador y le pasa por la entrada estándar la ruta del fichero de configuración	6. Si se encuentran campos sin llenar o no existen las rutas a los ficheros especificados se lanza en error, a través de la salida estándar, informándole al usuario las causas del error y pidiendo volver al paso 1. 7. Segmenta las imágenes 7.1. Si es activada la variable *save_ima_mask* es guardada la imagen en escala de grises 7.2. Si es activada la variable *save_binary_img* es guardada la imagen binaria 8. Extracción sus características 9. Pasa estas características a la Red Neuronal para obtener una categoría 10. Envía un lista, por la salida estándar, conformada por el nombre de las imágenes con su respectiva categoría
Prioridad:	Crítico

Cuadro 2.6: Descripción del CU Categorizar listas imágenes

Caso de Uso:	Probar el categorizador
Actores:	Usuario
	Continúa en la próxima página

<table>
<tr><td>Resumen:</td><td colspan="2">El Caso de Uso se inicia cuando el usuario decide entrenar el categorizador.</td></tr>
<tr><td>Referencia:</td><td colspan="2">R3</td></tr>
<tr><td>CU asociados:</td><td colspan="2">Categorizar lista de imágenes</td></tr>
<tr><td colspan="3">Flujo Normal de Eventos</td></tr>
<tr><td>Acción del Actor</td><td colspan="2">Respuesta del Sistema</td></tr>
<tr><td>1. Abre el fichero de configuración
2. Pone al categorizador en modo de prueba (mode = test)
3. Llena las variables referente a:
3.1. Sección de [test]
3.2. Sección de [segmentation]
3.3. Sección de [classifier_mlp]
4. Guarda los cambios realizados.
5. Invoca al clasificador y le pasa por la entrada estándar la ruta del fichero de configuración</td><td colspan="2">6. Si se encuentran campos sin llenar o no existen las rutas a los ficheros especificados se lanza en error, a través de la salida estándar, informándole al usuario las causas del error y pidiendo volver al paso 1.
7. Se llama al CU Categorizar lista de imágenes para obtener una lista de imágenes categorizadas. Para cada una de estas imágenes conociendo a priori el directorio al cual pertenece (e.j bening_imgs_path) y la categoría asignada por la clasificación (e.j pornography) se pueden obtener los falsos positivos y falsos negativos.
8. Envía un mensaje, por la salida estándar, con las estadísticas recogidas.</td></tr>
<tr><td>Prioridad:</td><td colspan="2">Crítico</td></tr>
</table>

Cuadro 2.7: Descripción del CU Probar el categorizador

2.10. Conclusiones

Después de definir las características de MCAIP se está en condiciones de efectuar su análisis y diseño a partir de las mismas. Además, dadas sus características distintivas, se concluye que el sistema propuesto podrá cumplir satisfactoriamente su labor dentro de MCADHTML.

Capítulo 3

Análisis y Diseño del sistema

3.1. Introducción

El objetivo principal del Análisis y Diseño es transformar los requisitos a una especificación que describa cómo implementar el sistema. En este capítulo, de MCAIP, se presentan: sus diagramas de clases del análisis, mostrando las clases participantes y relaciones entre ellas; sus diagramas de secuencia, evidenciando el orden de las acciones en los casos de uso cuando son invocados y, de sus clases del diseño, el diagrama y una descripción detallada. Además, se muestran los patrones utilizados para su diseño.

3.2. Análisis

Durante el análisis, los requisitos capturados en el capítulo anterior se especifican de manera más precisa y se estructuran de un modo que facilita su mantenimiento. Además se utiliza el lenguaje de los desarrolladores para describir los resultados, permitiendo razonar más sobre los aspectos internos del sistema.

3.2.1. Diagramas de Clases del Análisis

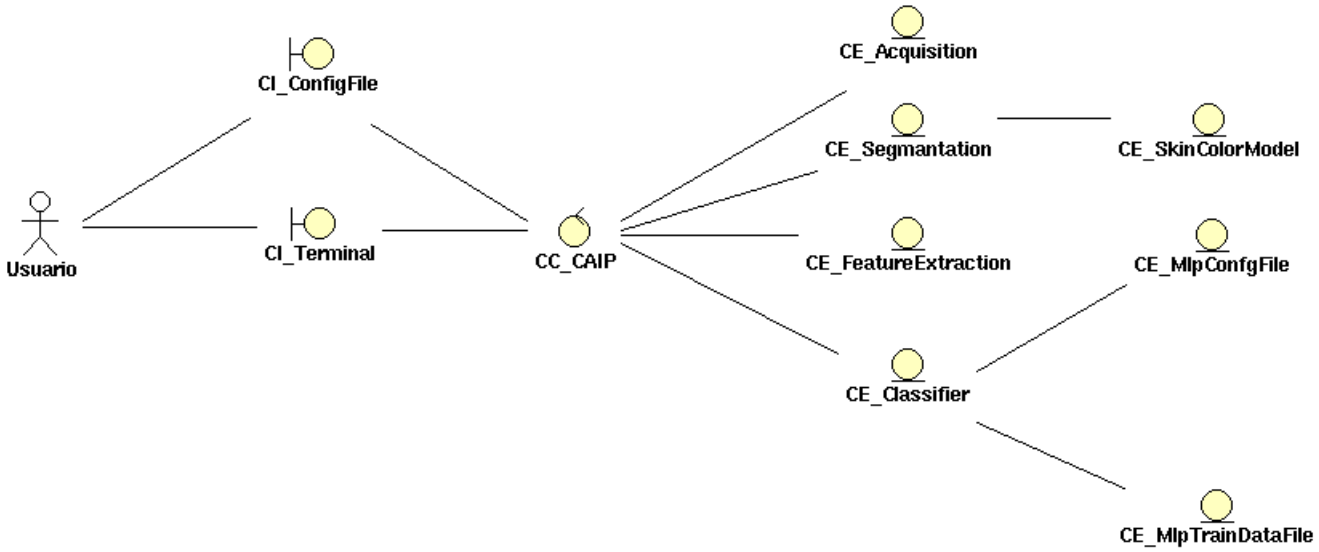

Figura 3.1: Diagrama de Clases del Análisis.

3.2.2. Diagramas de Secuencia del Análisis

El diagrama de secuencia de un sistema muestra gráficamente los eventos que fluyen de los actores al sistema. La creacion de 1os diagramas de la secuencia de un sistema forma parte de la investigación para conocer el sistema; se incluye, pues, dentro del modelo de análisis. El *Unified Modeling Language* (UML) ofrece una notacibn con 1os diagramas de la secuencia que muestran gráficamente 1os eventos que pasan de 1os actores a1 sistema.

Los diagramas de la secuencia de un sistema se preparan durante la fase de análisis de un ciclo de desarrollo. Su creacion depende de la formulacion previa de 1os casos de uso.

Antes de iniciar el diseño logico de como funcionaría una aplicacibn de software, es necesario investigar y definir su comportamiento como una “caja negra”. El comportamiento del sistema es una descripción de lo que hace, sin explicar la manera en que lo hace. Una parte de la descripcion es un diagrama de la secuencia del sistema.

El diagrama de la secuencia de un sistema deberia prepararse para el curso normal de 1os eventos de un caso de uso, teniendo en cuenta 1os cursos opcionales mas interesantes[12, pp. 135–137]. En las figuras 3.2, 3.3 y 3.4 se muestran los diagramas de secuencia de los CUs Entrenar el categorizador, Probar el categorizador y Categorizar lista de imágenes respectivamente.

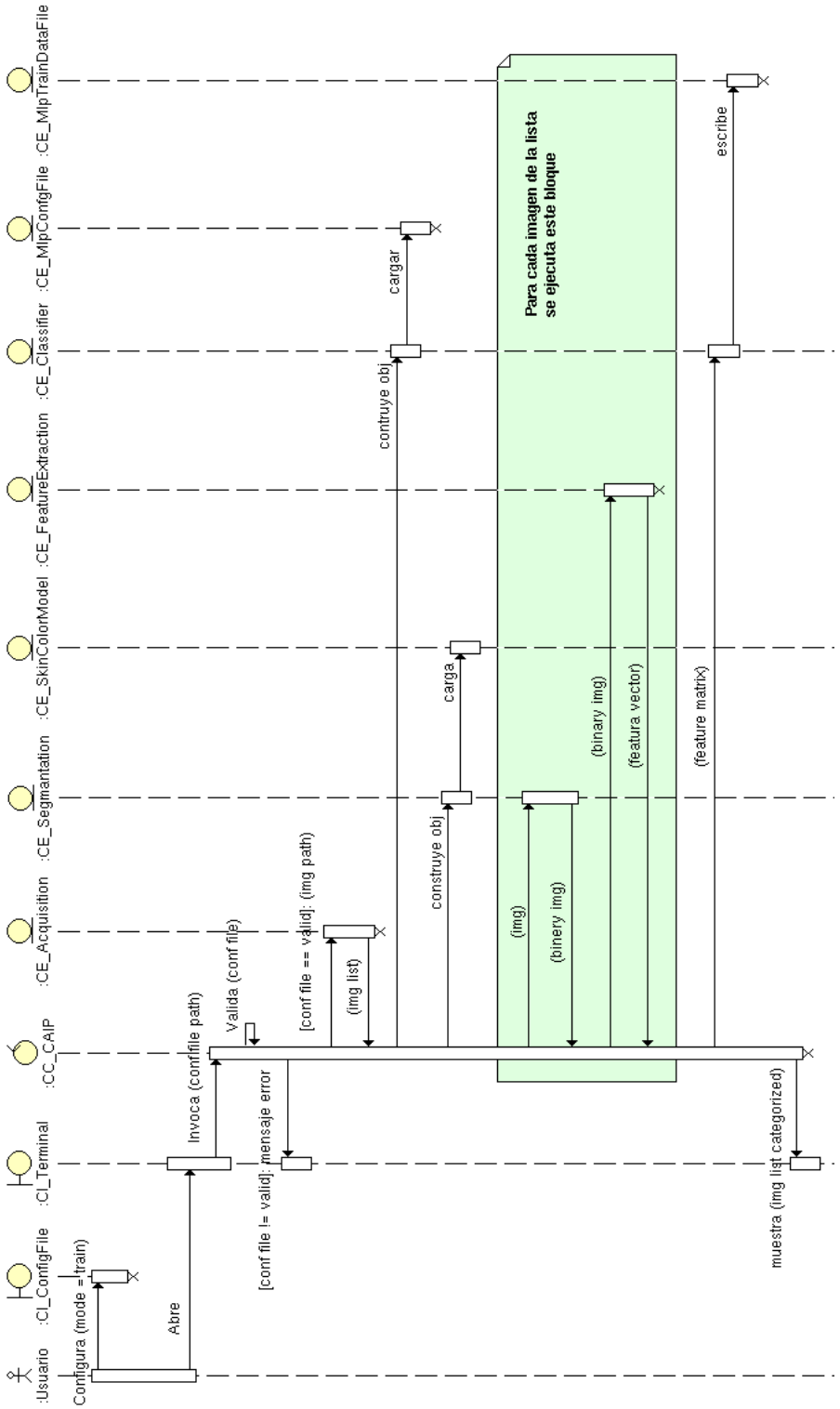

Figura 3.2: Diagrama de secuencia del análisis del CU Entrenar el categorizador.

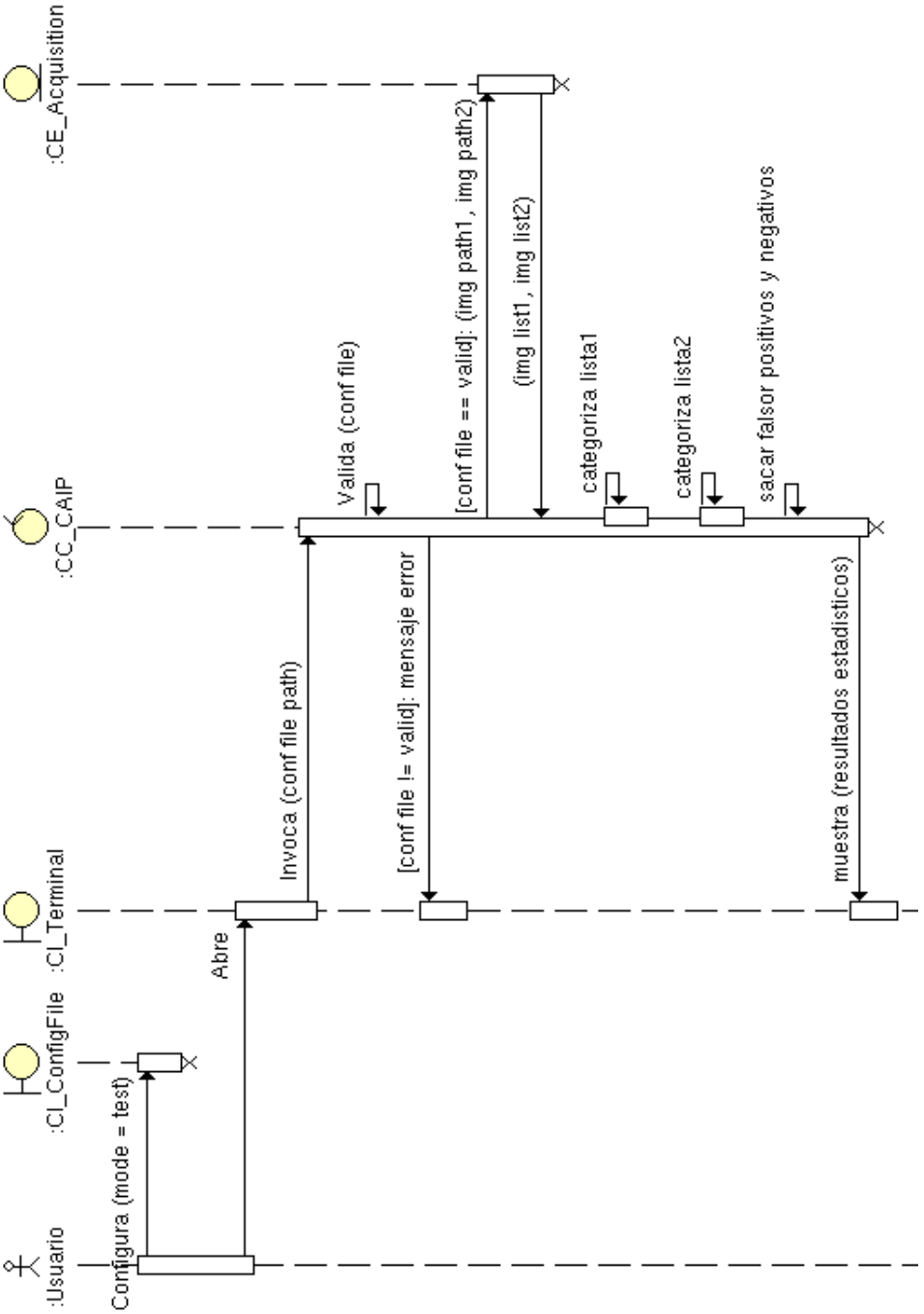

Figura 3.3: Diagrama de secuencia del análisis del CU Probar el categorizador.

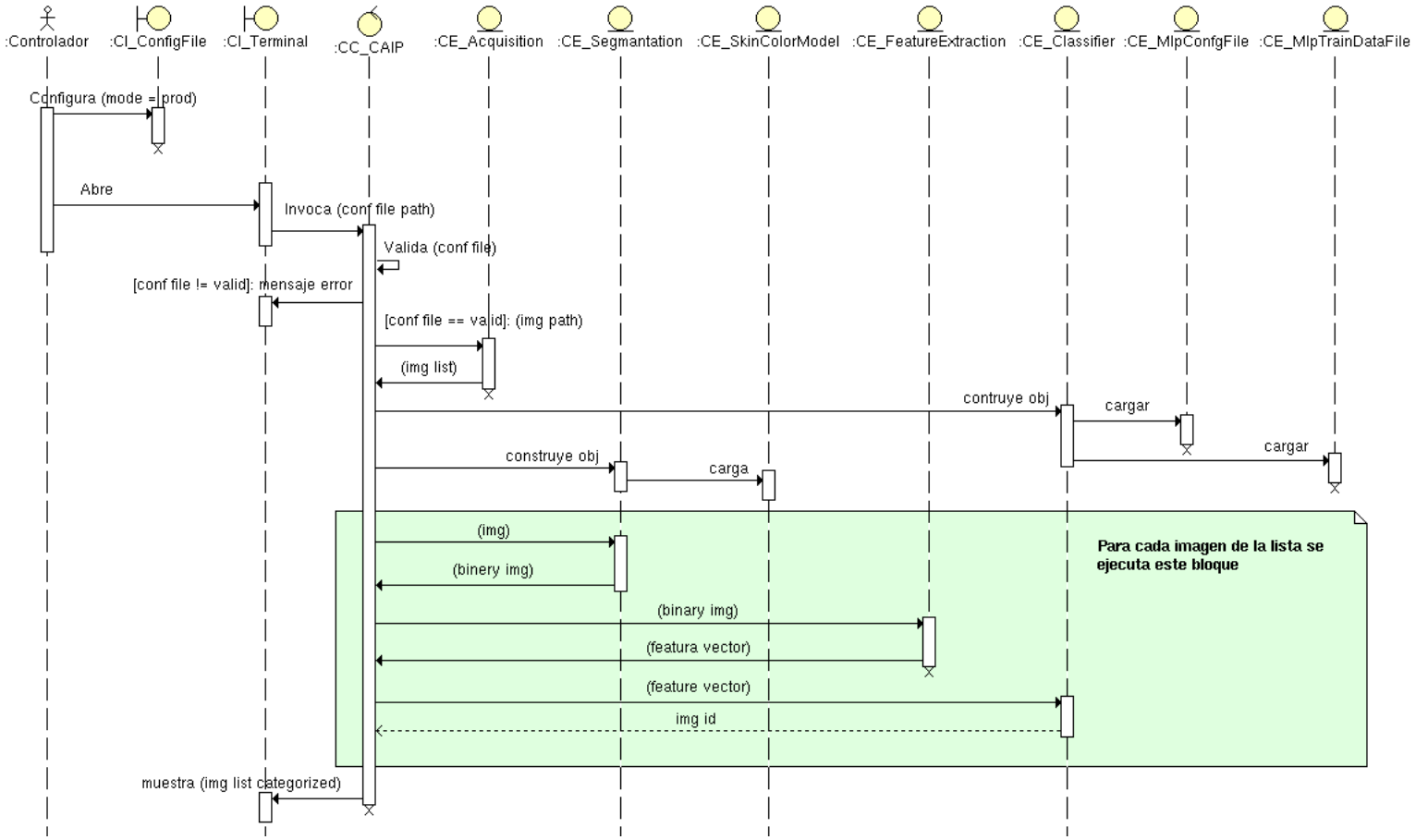

Figura 3.4: Diagrama de secuencia del analisis del CU Categorizar lista de imágenes.

3.3. Diseño

Durante el ciclo de desarrollo iterativo es posible pasar a la fase de diseiio, una vez terminados estos documentos del analisis. Durante este paso se logra una solucion logica que se funda en el paradigma orientado a objetos. Su esencia es la elaboracion de dia gramas de interaccih, que muestran grhficamente como 10s objetos se comunicarhn entre ellos a fin de cumplir con 10srequerimientos.

El advenimiento de 10s diagramas de interaccion nos permite dibujar diagramas de diseiio de clases que resumen la definicion de las clases (e interfaces) implementables en software.

En 10s siguientes secciones examinaremos la creacion de estos artefactos. Los diagramas de interaccion son 10s mas importantes de ellos (desde el punto de vista de la preparacion de un buen diseiio) y exigen gran dedicacion y esfuerzo creativos. Para prepararlos hay que aplicar 10s principios de la asignacion de responsabilidadesy utilizar 10s patrones de diseiio. Por tanto, en 10s siguientes capitulos nos centraremos en esos principios y patrones del diseiio orientado a objetos.

La creacion de 10s diagramas de interaccion exige conocer:

- Los principios de la asignacion de responsabilidades.
- Los patrones del diseiio.

3.3.1. Diagramas de Secuencia del Diseño

Al igual que en el anális los diagramas de secuencia, o interacción en general, constituyen uno de los artefactos más importantes que se generan.

El tiempo y esfuerzo dedicados a su preparación debería absorver un porcentaje considerable de la actividad totol desegnada al poryecto.

Para mejorar la calidad de su diseño, es posible aplicar patrones, principios y expresiones codificadas.

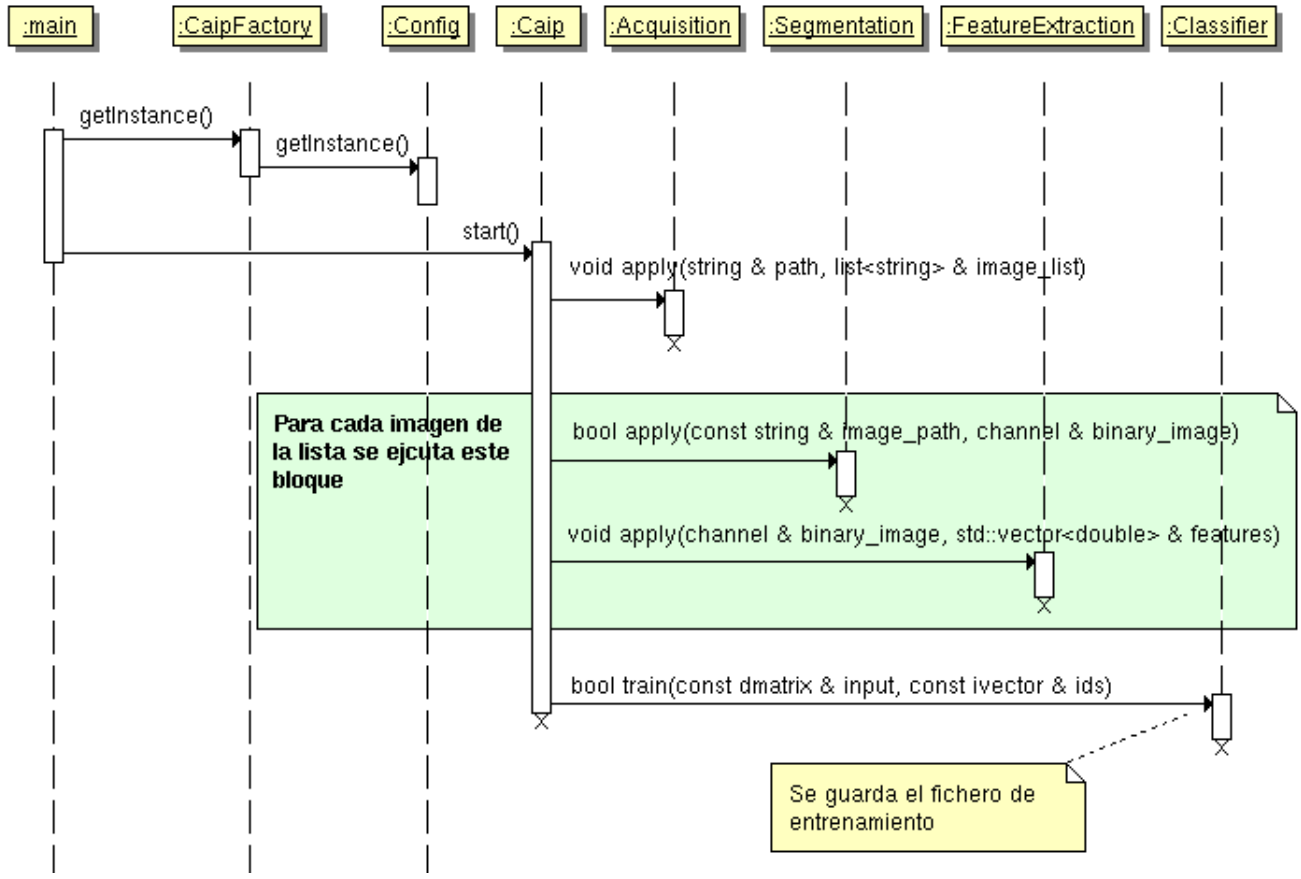

Figura 3.5: Diagrama de secuencia del diseño del CU Entrenar el categorizador.

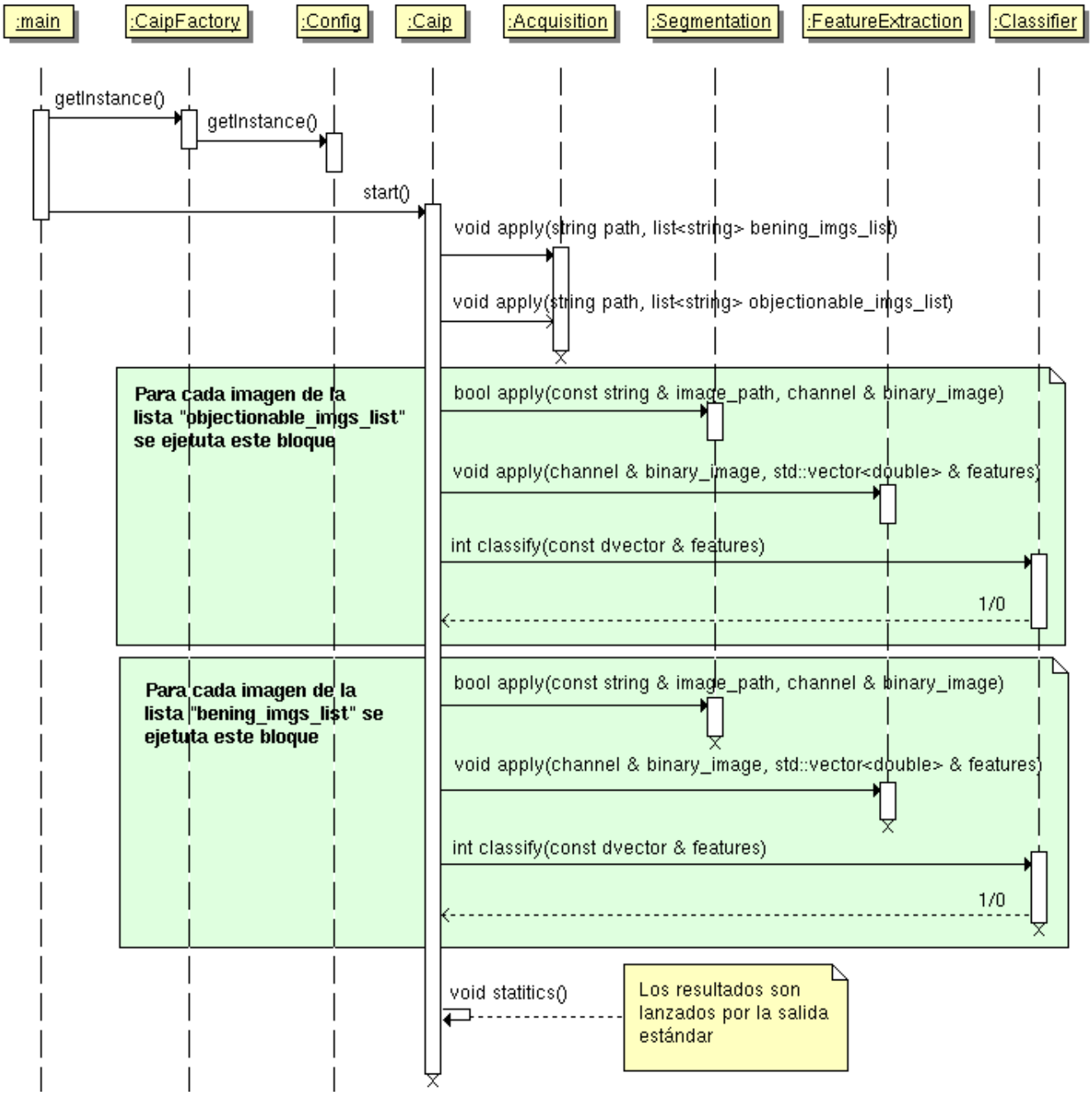

Figura 3.6: Diagrama de secuencia del diseño del CU Probar el categorizador.

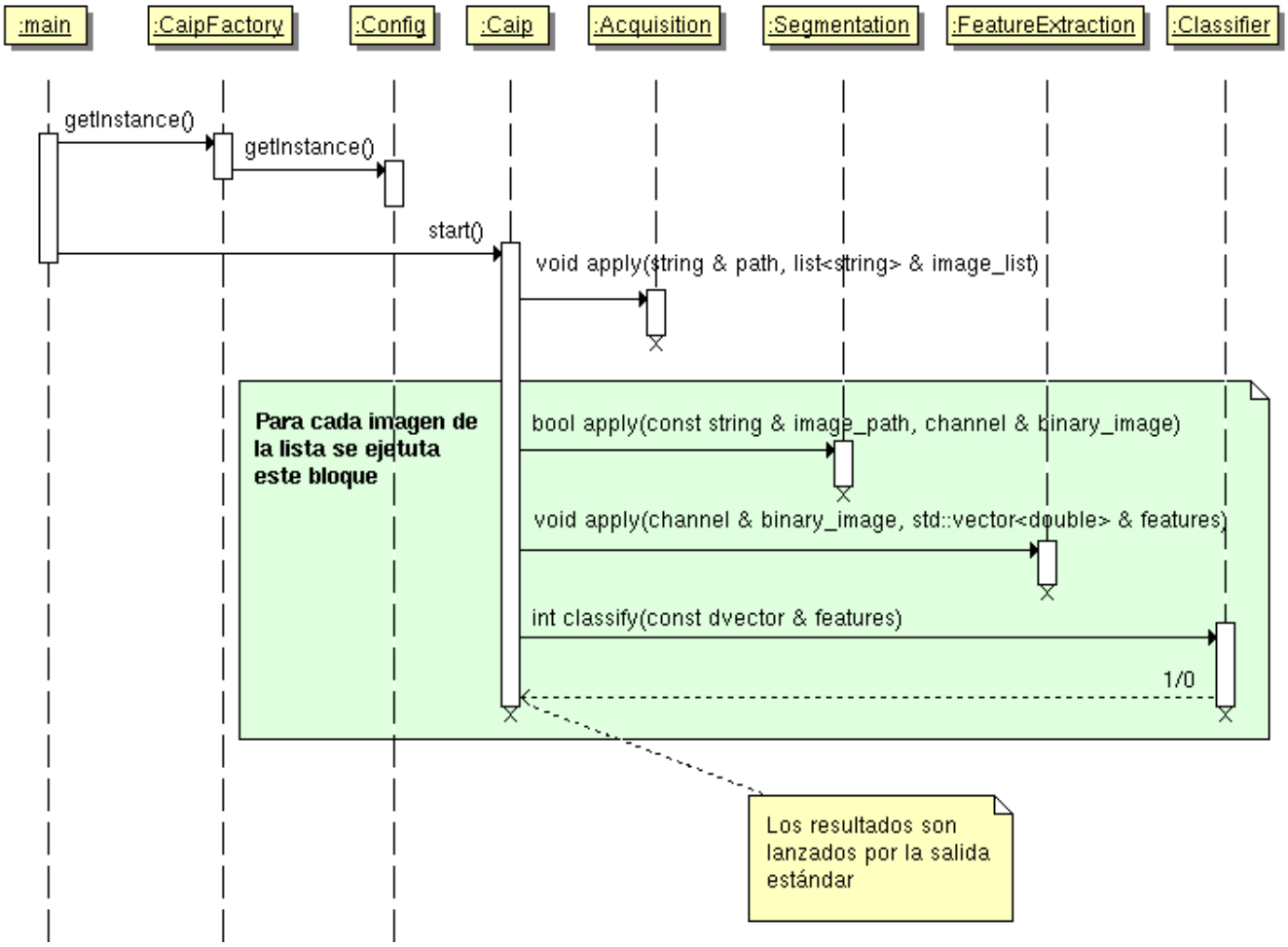

Figura 3.7: Diagrama de secuencia del diseño del CU Categorizar lista de imágenes.

3.3.2. Patrones de diseño

Un sistema orientado a objetos se compone de objetos que envían mensajes a otros objetos para que lleven a cabo las operaciones. Los diagramas de interacción describen gráficamente la solución a partir de 1os objetos en interacción que estas responsabilidades y poscondiciones satisfacen.

La calidad de diseño de la interacción de 1os objetos y la asignación de responsabilidades presentan gran variación. Las decisiones poco acertadas dan origen a sistemas y componentes frágiles y difíciles de mantener, entender, reutilizar o extender. Una implementación hábil se funda en 1os principios cardinales que rigen un buen diseño orientado a objetos. En 1os patrones GRASP se codifican algunos de ellos, que se aplican a1 preparar 1os diagramas de interaccion, cuando se asignan las responsabilidades o durante

ambas actividades[12, p. 185]. Para el diseño de la solución al problema presentado en este trabajo se utilizaron los siguientes patrones.

GRASP: Patrones para asignar responsabilidades

Estos patrones se aplican durante la elaboración de 1os diagramas de interacción, a1 asignar las responsabilidades a los objetos y al diseñar la colaboración entre ellos[12, pp. 191-211].

Polimorfismo Cuando varia el tipo (clase) de alternativas o comportamientos relacionados, asigna la responsabilidad del comportamiento mediante operaciones polimórficas a los tipos en que varia el comportamiento, en este coso la clase Caip tiene tres tipo de comportamiento (Prod, Test, Train), por lo que se implementan tres clases con características similares pero comportamiento diferente para una acción dada (CaipProd, CaipTest y CaipTrain).

Experto asigna una responsabilidad al experto en información, o sea, la clase que cuenta con la información necesaria para cumplir la responsabilidad, en este caso:

- Caip tiene la responsabilidad de hacer el flujo de categorización completo (Adquisición, Segmentación, Extracción de características, Entrenamiento y Categorización).
- CaipFactory tiene la responsabilidad de crear y devolver un objeto de tipo CaipProd, CaipTest o CaipTrain.
- Adquisición tiene la responsabilidad de devolver un lista de string (que representa la lista de imágenes en un directorio).
- Segmentation tienen la responsabilidad de devolver una imagen binaria.
- FeatureExtraction tiene la responsabilidad de devolver el vector de características de la imagen.
- Classifier tiene la responsabilidad de de entrenar o categorizar, en dependencia del modo en que se encuentre el sistema (train, prod, test).

Creador asigna a la clase B la responsabilidad de crear una instancia de clase A, en este caso:

- Caip crea una instancia de Adquisición
- Caip crea una instancia de Segmentation

- Caip crea una instancia de FeatureExtraction
- Caip crea una instancia de Classifier
- Adquisición crea una instancia de Directory
- Segmentation crea una instancia de SkinColorModel

Controlador asigna la responsabilidad del manejo de un mensaje de los eventos de un sistema a una clase, en este caso corresponde a la clase Caip esta responsabilidad.

OODP: Patrones para el diseño orientado a objeto

Singleton A veces conviene permitir la visibilidad global o un solo punto de acceso a una instancia individual de una clase y no alguna otra forma de visibilidad, esto ocurre en el caso de la instancia de la clase Config, ya que es utilizada por las clases Caip, Adquisición, Segmentation y Classifier siendo el mismo objeto.

Factory ayuda a modelar una interfaz la cual a la hora de crearse un objeto decide que clase instanciar. Llamamos a este patrón Factory porque el es el responsable de fabricar un objeto. Ayuda a instanciar la clase apropiada mediante la creación del objeto correcto de un grupo de clases relacionadas. Este patrón favorece al bajo acoplamiento eliminando la necesidad de crear manualmente en el código el objeto que se requiere[16]. Esto pasa con las clases hijas de Caip, en dependencia del modo en que se ejecute el sistema (Prod, Test o Train) la clase CaipFactory devolverá el objeto correspondiente.

3.3.3. Diagrama de Clases del Diseño

Un Diagrama de Clases del Diseño describe gráficamente las especificaciones de las clases de un sistema y es una representación más concreta que los Diagramas de Clases del Análisis. Contiene información sobre clases, asociaciones, atributos y métodos. En las figuras siguientes se presentan el Diagrama de Clases del Diseño del Módulo MCAIP:

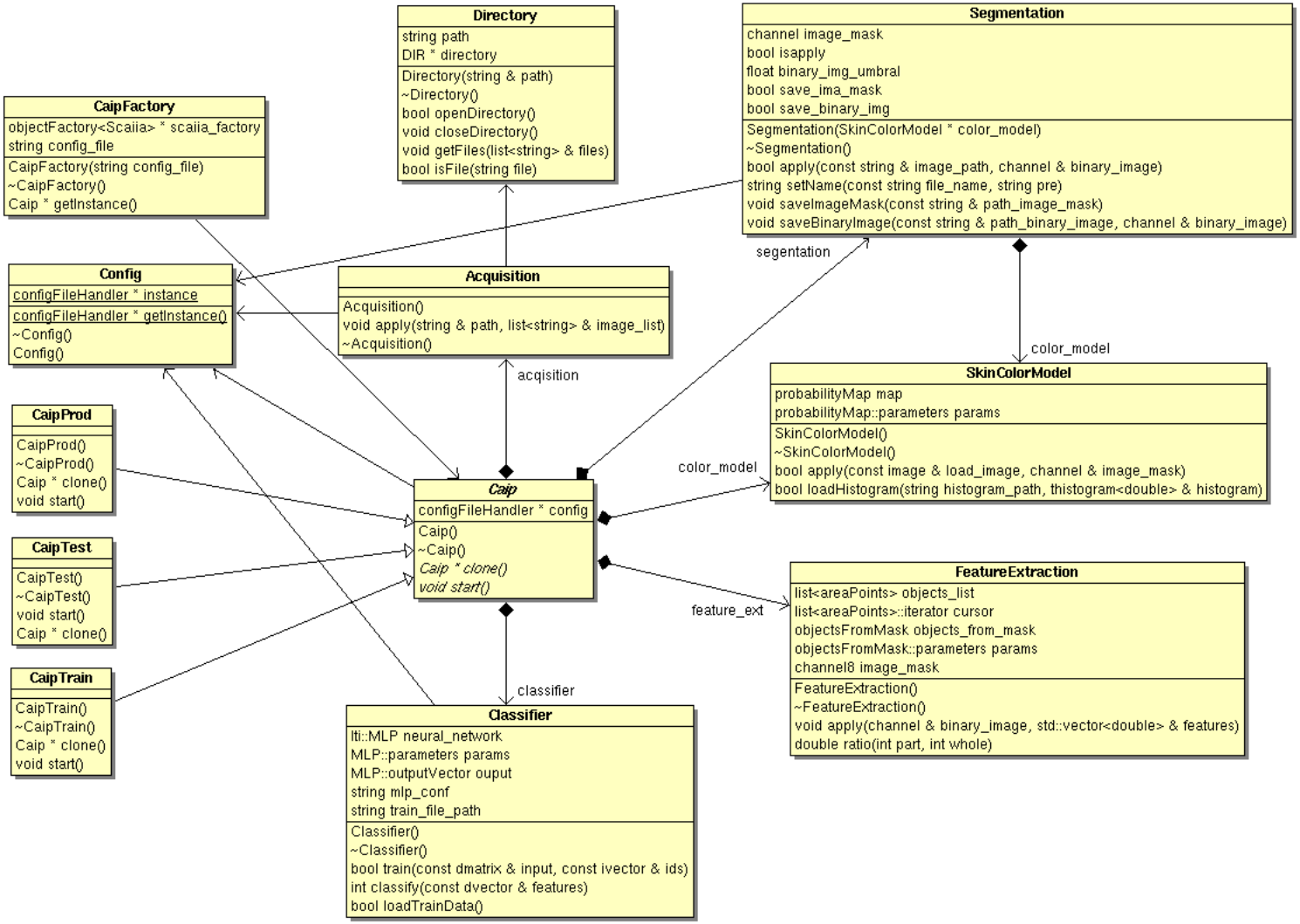

Figura 3.8: Diagrama de de Clases del Diseño.

3.3.4. Descripción de las Clases

A continuación se describen las clases del diseño del Módulo MCAIP:

Nombre	Caip
Tipo de clase	Controladora
Atributos	
Nombre	Tipo
conf	configFileHandler
Responsabilidades	
Nombre: Caip()	
Descripción: Constructor de la clase, debuelve un objeto **Caip**.	
Nombre: ∼Caip()	
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.	
Nombre: void start()	
Descripción: Médoto abstracto que se implementará en las clases hijas.	

Cuadro 3.1: Descripción de la Clase Caip

Nombre	Config
Tipo de clase	Entidad
Atributos	
Nombre	Tipo
instance	configFileHandler*
Responsabilidades	
Nombre: Config()	
Continúa en la próxima página	

Descripción: Constructor de la clase, debuelve un objeto **Config**.
Nombre: ∼Config()
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.
Nombre: configFileHandler* Config::getInstance()
Descripción: Debuelbe una instancia única del atributo *instance*.

Cuadro 3.2: Descripción de la Clase Config

Nombre	Acquisition
Tipo de clase	Entidad
Responsabilidades	
Nombre: Acquisition()	
Descripción: Constructor de la clase, debuelve un objeto **Acquisition**.	
Nombre: ∼Acquisition()	
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.	
Nombre: void apply(string & path, list<string> image_list)	
Descripción: El método recibe la ruta de un directorio y debuelbe una lista de stringes con el nombre completo (incluyendo la ruta e.j la ruta pasaes es /home/filpacon y el nombre de la imagen debuelta es /home/filpacon/filp.png) de todas las imágenes del directorio	

Cuadro 3.3: Descripción de la Clase Acquisition

Nombre	Directory
Tipo de clase	Entidad
Atributos	
Continúa en la próxima página	

Nombre	Tipo
path	string
directory	DIR
Responsabilidades	
Nombre: Directory(string path)	
Descripción: Constructor de la clase, debuelve un objeto **Directory**.	
Nombre: ∼Directory()	
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.	
Nombre: bool openDirectory()	
Descripción: Abre el directorio espesificada en caso de ser un directorio.	
Nombre: void closeDirectory()	
Descripción: Sierra el directorio si ha sido abierto.	
Nombre: isFile(string file)	
Descripción: debuelve verdadero de la ruta entrada pertenece a un fichero (no a una carpeta).	
Nombre: getFiles(list<string>& files)	
Descripción: Debuelve el la variables *file* pasada por referencia la lista de los ficheros que se encuentran en el directorio (estos fichero deben de ser imágenes)	

Cuadro 3.4: Descripción de la Clase Directory

Nombre	Segmentation
Tipo de clase	Entidad
Atributos	
Nombre	Tipo
image_mask	channel
isapply	bool
Continúa en la próxima página	

binary_img_umbral	float
save_img_mask	bool
save_binary_img	bool
Responsabilidades	
Nombre: Segmentation(SkinColorModel* color_model)	
Descripción: Constructor de la clase, debuelve un objeto **Segmentation**.	
Nombre: ∼()Segmentation	
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.	
Nombre: void saveBinaryImage(const string & path_binary_image, channel & binary_image)	
Descripción: Salva la imagen segmentada (binaria) si la variables del fichero de configuración *save_binary_img* es igual a 1	
Nombre: void saveImageMask(const string & path_image_mask, channel & image_mask)	
Descripción: Salva la mascara de la imagen (escala de grises) si la variables del fichero de configuración *save_img_mask* es igual a 1	
Nombre: string setName(const string file_name, string pre)	
Descripción: Cambia el nombre del fichero en la variables *file_name*, poniendole un prefijo pasado *pre* (e.j si el fichero entrado es /home/filpacon/img.png y el prefijo es 10_ entonces la nueva ruta quedaría de la siguiente forma /home/filpacon/10_img.png)	
Nombre: bool apply(const string & image_path, channel & binary_image	
Descripción: Segmenta la imagen que está en la ruta espesificada. Este proceso de segmentación na es más que convertir la imágen dada en una matriz de unos y ceros representando piel y no piel respectivamente.	

Cuadro 3.5: Descripción de la Clase Segmentation

Nombre	SkinColorModel
Tipo de clase	Entidad
Continúa en la próxima página	

Atributos	
Nombre	Tipo
map	ProbabilityMap
params	ProbabilityMap::parameters
Responsabilidades	
Nombre: SkinColorModel()	
Descripción: Constructor de la clase, debuelve un objeto **SkinColorModel**.	
Nombre: ∼SkinColorModel()	
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.	
Nombre: bool loadHistogram(string histogram_path, thistogram<double> & histogram)	
Descripción: Falta	
Nombre: bool apply(const image & load_image, channel & image_mask)	
Descripción: Falta	

Cuadro 3.6: Descripción de la Clase SkinColorModel

Nombre	FeatureExtraction
Tipo de clase	Controladora, Entidad
Atributos	
Nombre	Tipo
objects_list	list<areaPoints>
cursor	list<areaPoints>::iterator
objects_from_mask	objectsFromMask
params	objectsFromMask::parameters
image_mask	channel8
Responsabilidades	
Continúa en la próxima página	

Nombre: FeatureExtraction()
Descripción: Constructor de la clase, debuelve un objeto **FeatureExtraction**.
Nombre: ∼FeatureExtraction()
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.
Nombre: double ratio(int part, int whole
Descripción: Debulve la razón entre *part* y *whole* si ambas son mayores que cero.
Nombre: void apply(channel & binary_image, std::vector<double> & features)
Descripción: Dada una imagen binaria se le extraen las características previamente definidas y se devuelven en forma de vector el la variable *features* pasada por referencia.
Nombre:
Descripción:

Cuadro 3.7: Descripción de la Clase FeatureExtraction

Nombre	Classifier
Tipo de clase	Controladora, Entidad
Atributos	
Nombre	Tipo
neural_network	lti::MLP
params	MLP::parameters
ouput	MLP::outputVector
mlp_conf	string
train_file_path	string
Responsabilidades	
Nombre: ()Classifier	
Descripción: Constructor de la clase, debuelve un objeto **Classifier**.	
Continúa en la próxima página	

Nombre: ∼Classifier()
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.
Nombre: bool loadTrainData()
Descripción: Carga el fichero de entrenamiento a la hora de categorizar
Nombre: int classify(const dvector & features)
Descripción: Devuelve el *ids* al que pertenece la imagen correspondiente al vector característico pasado por parametro (1 es objectionable_image y 0 bening_image)
Nombre: bool train(const dmatrix & input, const ivector & ids)
Descripción: Entrena el categorizador. Por parametro es pasada una matriz con los vectores da características, en forma de filas, de todas las imágenes de la colección de entrenamiento, así como un vector indicanto a que clase pertenece cada una de las filas

Cuadro 3.8: Descripción de la Clase Classifier

Nombre	CaipTrain
Tipo de clase	Entidad
Responsabilidades	
Nombre: CaipTrain()	
Descripción: Constructor de la clase, debuelve un objeto **CaipTrain**.	
Nombre: ∼CaipTrain()	
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.	
Nombre: void start()	
Continúa en la próxima página	

Descripción: Entrena el clasificador. Este método es el responsable de extraerle el vector de características a cada imagen de la colección de entrenamiento, conformando una matriz; formar el vector de clases que contendrá la clase a la que pertenece cada imagen. Esta matriz y vector son pasados el clasificador y este entrena una red neuronal. El modelo de categorización (entrenamiento) es guardado en un fichero para ser leido a la hora de clasificar. El finalizar se lanza por la salida estándar un mensaje indicando la terminación del entrenamiento.

Cuadro 3.9: Descripción de la Clase CaipTrain

Nombre	CaipProd
Tipo de clase	Entidad
Responsabilidades	
Nombre: CaipTrain()	
Descripción: Constructor de la clase, debuelve un objeto **CaipProd**.	
Nombre: ∼CaipTrain()	
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.	
Nombre: void start()	
Descripción: Categoriza una lista de imagenes. La ruta de las imágenes a categorizar se intorducen mediante al fichero de configuración. El resultado se lanza a través de la salida estándar y su formatos es: *nombre_imagen* -> *clase_pertenece*, esto para cada una de las imágenes de la lista.	

Cuadro 3.10: Descripción de la Clase CaipProd

Nombre	CaipTest
Tipo de clase	Entidad
Continúa en la próxima página	

Responsabilidades
Nombre: CaipTrain()
Descripción: Constructor de la clase, debuelve un objeto **CaipTest**.
Nombre: ∼CaipTrain()
Descripción: Destructor de la clase, borra todas las variables dinamicas del objeto.
Nombre: void start()
Descripción: Prueba el categorizador. dado dos rutas, una que contiene imágenes pornográficas y otra no pornográficas, se clasifican por separado y al tener conocimiento a priori a que clase pertenece una imagen, es fácil determinar los falsesos positivos y falsesos negativos. Estos resultados estodisticos son lanzados por la salida estándar.

Cuadro 3.11: Descripción de la Clase CaipTest

3.4. Implementación

Culminado el diseño del software, habiéndose generadorado los artefactos necesarios, se sigue con su implementación para lo cual hace falta generar algunos artefactos que guiarán dicha implementación. Los principales artefactos de la implementación son el diagrama de componentes y el diagrama de despliegue, ambos son mostrados a continuación.

3.4.1. Diagrama de componentes

Los diagramas de componentes muestran las dependencias del compilador y del runtime.[en]tre 1os componentes del software; por ejemplo, los archivos del código fuente y la bibliotecas[12, p. 429]. En la figura 3.9 se muestra es diagrama de componentes del Módulo de Categorización Automática de Imágenes Pornográficas. Se puede apreciar la fuerte dependencia que tiene el software con la biblioteca stdlib así como la biblioteca LTI-Lib; también se muestra la dependencia del software con un programa terminal desde el cual se ejecutará, este programa terminal sirve como una interfaz entre el Usuario/Controlador y el MCAIP.

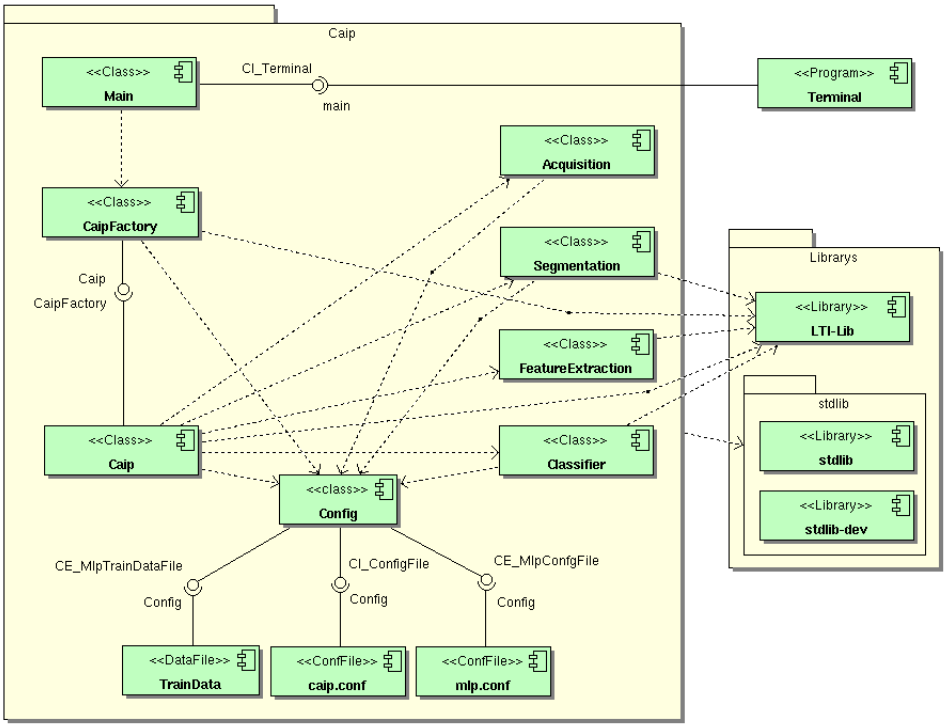

Figura 3.9: Diagrama de componentes.

3.4.2. Diagrama de despliegue

Los diagramas de despliegue muestran a los nodos procesadores la distribución de los procesos y de los componentes[12, p. 430]. En la figura 3.10 se muestra el diagrama de despliegue del Módulo de Categorización Automática de Imágenes Pornográficas. Dicha figura ilustra los componentes necesarios para el funcionamiento del software en un ordenador. Los componentes mostrados son suficientes para la ejecución del software, sin embargo para su compilación; en coso de algún cambio en su código fuente, es necesario también tener instalada la biblioteca LTI-Lib.

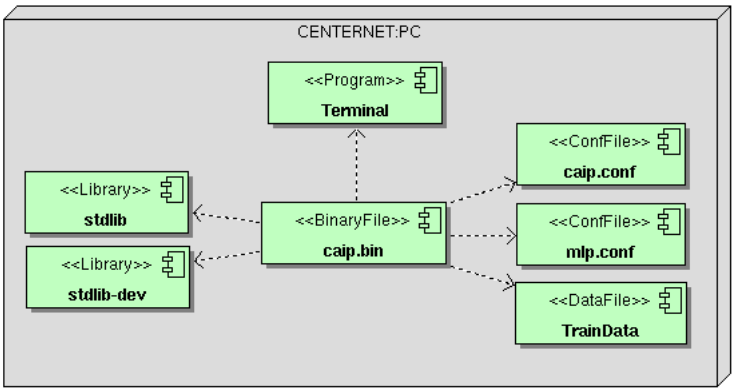

Figura 3.10: Diagrama de despliegue.

3.5. Conclusiones

Tras efectuar el Análisis de MCAIP se comprendieron sus requisitos de manera más precisa, analizándolos con mayor profundidad. Además, se consiguió preparar y simplificar las subsiguientes actividades del diseño. Respecto a los diagramas de clases del análisis que se obtuvieron, se concluye que pueden ser aplicados a varios diseños y cada una de sus clases de control, entidad e interfaz aislarán futuros cambios al comportamiento e información que representan.

Llevando a cabo el Diseño de MCAIP se encontró su forma y se obtuvo un sistema flexible a los cambios en los requisitos. Mediante los diagramas de secuencia se mostró la forma en que los objetos se comunican entre sí al transcurrir el tiempo, contribuyendo de manera importante a entender el comportamiento del sistema. Se evidenció que el lenguaje de programación seleccionado (C++) provee mecanismos para tratar adecuadamente los errores y brindar la ayuda necesaria respecto al uso del sistema. La habilidad más importante en el análisis y diseño orientados a objetos, asignar eficientemente las responsabilidades a los componentes del software, se cumplió satisfactoriamente.

Para la ejecución de la implementación se generaron los diagramas de componentes y despliegue que sirvieron de guia, dando una mejor idea de las relaciones entre los componentes de software. Esto permitió una correcta implementación de los componentes de forma separada.

Conclusiones

Con la culminación del presente trabajo de diploma se cumple con el objetivo trazado en el mismo, mediante el diseño e implementación de un sistema capaz de categorizar de forma automática imágenes pornográficas. Se cumplió además con los siguientes aspectos:

- Se aplicaron patrones de diseño para obtener una arquitectura de clases robusta y a la ves flexible a cambios.
- Se aplicó el modelo de john y rehg para la segmentación de la imagen obteniendo buenos resultados.
- Se obtuvieron siete descriptores de las imágenes pornográficas que sirvieron para su categorización.
- Se eligió como algoritmo de entrenamiento y categorización un Perceptrón multicapa, el cual es lo suficientemente eficiente para resolver este problema.

Recomendaciones

Para mejorar la arquitectura del software se recomienda:

- Estudiar y aplicar el patrón de diseño *filter* el cual mejorarará la arquitectura del proceso de categorización.
- Definir un directorio, el cual puede ser **/$HOME/.CAIP**, para guardar el los ficheros de configuración del software y de esta manera evitar la introducción de estos por la entrada estándar.

Para mejorar el proceso de clasificación se recomienda:

- Estudiar y aplicar el método Otsu para la selección automática del umbral a utilizar.
- Detectar la presencia de rostro en la imagen y tomar esto como una característica.
- Estudiar otras posibles característica de las imágenes pornográficas.
- Elegir una adecuada colección de entrenamiento basada el las características.
- Entrenar el sistema con una adecuada colección de entrenamiento.
- Correr el sistema en modo de prueba, con al menos 1000 imágenes de ambas categorías, para comprobar su eficiencia y eficacia.

Referencias

[1] IDC. *The Expanding Digital Universe* [Internet]. Disponible en: <http://www.emc.com/collateral/analyst-reports/expanding-digital-idc-white-paper.pdf> [Adcedida 22 junio 2008].

[2] INFOBAE. *Qué tan grande es internet en la actualidad* [Internet]. Disponible en: <http://www.infobae.com/contenidos/305016-100796-0-Qu%C3%A9-tan-grande-es-internet-la-actualidad> [Adcedida 22 junio 2008].

[3] TopTenREVIEWS. *Internet Filter Software Review 2008* [Internet]. Disponible en: <http://internet-filter-review.toptenreviews.com/index.html#anchor> [Adcedida 20 mayo 2008].

[4] LTU technologies. *LTU Technologies: image search and image filter software* [Internet]. Disponible en: <http://www.ltutech.com/en/technology-and-products.image-filter.html> [Adcedida 13 junio 2008].

[5] POESIA project. *Poesia Project* [Internet]. Disponible en: <http://www.poesia-filter.org> [Adcedida 14 junio 2008].

[6] José Carlos Cortizo Pérez y Diego Expósito Gil. *Deteccion de Imágenes Pornográficas mediante Redes Neuronales Artificiales* [Internet]. Disponible en: <http://www.esp.uem.es/jccortizo/RNAs.pdf> [Adcedida 13 junio 2008].

[7] Luis Alonso Romero and Teodoro Calonge Cano. *Redes Neuronales y Reconocimiento de Patrones* [Internet]. Disponible en: <http://lisisu02.fis.usal.es/{~}airene/capit1.pdf> [Adcedida 14 junio 2008].

[8] Tinku Acharya y Ajoy K. Ray. *IMAGE PROCESSING Principles and Applications*. John Wiley and & Sons, 2005.

[9] TIOBE. *TIOBE Programming Community Index for June 2008* [Internet]. Disponible en: <http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html> [Adcedida 14 junio 2008].

[10] Ivar Jacobson et al. *El Proceso Unificado de Desarrollo de Software*. Addison Wesley Publishing Company, Noviembre de 2000, pp 30-34.

[11] *Modelo de Dominio* [Internet]. Disponible en: <http://iie.fing.edu.uy/ense/asign/desasoft/practico/hoja8/ejemplos_clase2.pdf> [Adcedida 20 mayo 2008].

[12] Craig Larman. *UML y Patrones: Introducción al análisis y diseño orientado a objetos*. Addison Wesley Longman.

[13] dzoom. *Formatos de almacenamiento de imagenes digitales* [Internet]. Disponible en: <http://www.dzoom.org.es/cont-38-formato-fichero-imagen-fotografia.html> [Adcedida 20 junio 2008].

[14] Darío de Miguel Benito y Juan José Pantrigo Fernández. *Detección automática del color de la piel en imágenes bidimencionales basado en el análisis de reguiones*. Universidad Rey Juan Carlos, España 2005. Disponible en: <http://www.escet.urjc.es/~jjpantrigo/PFCs/MemoriaPielFeb05.pdf> [Adcedida 20 junio 2008].

[15] *Statitical Color Models with Application to Skin Detection*. Dicember 1998, Cambridge Research Laboratory.

[16] Gopalan Suresh Raj. *The Factory Method Design Pattern* [Internet]. Disponible en: <http://gsraj.tripod.com/design/creational/factory/factory.html> [Adcedida 21 junio 2008].

Bibliografía

- *The Harvard Style of referencing published material*. Disponible en: <http://skillsforlearning.leedsmet.ac.uk/harvard_2004.pdf> [Adcedida 28 junio 2008].
- Joaquín Ataz López. *Guia casi completa de* BiBTeX. Disponible en: <http://www.ctan.org/tex-archive/info/spanish/guia-bibtex/guia-bibtex.pdf> [Adcedida 28 junio 2008].
- Tobias Oetiker. *An Acronym Environment for* LaTeX. Disponible en: <http://www.ctan.org/tex-archive/macros/latex/contrib/acronym/acronym.pdf> [Adcedida 28 junio 2008].
- Tobias Oetiker, Irene Hyna, Elisabeth Schlegl, Hubert Partl. *The Not So Short Introduction to* LaTeX. Disponible en: <http://tobi.oetiker.ch/lshort/lshort.pdf> [Adcedida 28 junio 2008].
- Raúl Mata Botana. *Tablas en* LaTeX. Disponible en: <http://www.lug.fi.uba.ar/documentos/tablas/tablas.pdf> [Adcedida 28 junio 2008].
- George Grätzer. *Math into* LaTeX. Disponible en: <ftp://ftp.tex.ac.uk/tex-archive/info/mil/mil.pdf> [Adcedida 28 junio 2008].
- Vice-Rectorí de Formación Departamento de Práctica Profecional. *Propuesta de Guía para la presentación del Trabajo de Diploma*. 2005-2006.
- IDC. *The Expanding Digital Universe* [Internet]. Disponible en: <http://www.emc.com/collateral/analyst-reports/expanding-digital-idc-white-paper.pdf> [Adcedida 22 junio 2008].
- INFOBAE. *Qué tan grande es internet* [Internet]. Disponible en: <http://www.infobae.com/contenidos/305016-100796-0-Qu%C3%A9-tan-grande-es-internet-la-actualidad> [Adcedida 22 junio 2008].

- TopTenREVIEWS. *Internet Filter Software Review 2008* [Internet]. Disponible en: <http://internet-filter-review.toptenreviews.com/index.html#anchor> [Adcedida 20 mayo 2008].
- LTU technologies. *LTU Technologies: image search and image filter software* [Internet]. Disponible en: <http://www.ltutech.com/en/technology-and-products.image-filter.html> [Adcedida 13 junio 2008].
- POESIA project. *Poesia Project* [Internet]. Disponible en: <http://www.poesia-filter.org> [Adcedida 14 junio 2008].
- José Carlos Cortizo Pérez y Diego Expósito Gil. *Deteccion de Imágenes Pornográficas mediante Redes Neuronales Artificiales* [Internet]. Disponible en: <http://www.esp.uem.es/jccortizo/RNAs.pdf> [Adcedida 13 junio 2008].
- Luis Alonso Romero and Teodoro Calonge Cano. *Redes Neuronales y Reconocimiento de Patrones* [Internet]. Disponible en: <http://lisisu02.fis.usal.es/{~}airene/capit1.pdf> [Adcedida 14 junio 2008].
- Tinku Acharya y Ajoy K. Ray. *IMAGE PROCESSING Principles and Applications*. John Wiley and & Sons, 2005.
- TIOBE. *TIOBE Programming Community Index for June 2008* [Internet]. Disponible en: <http://www.tiobe.com/index.php/content/paperinfo/tpci/index.html> [Adcedida 14 junio 2008].
- Ivar Jacobson et al. *El Proceso Unificado de Desarrollo de Software*. Addison Wesley Publishing Company, Noviembre de 2000, pp 30-34.
- *Modelo de Dominio* [Internet]. Disponible en: <http://iie.fing.edu.uy/ense/asign/desasoft/practico/hoja8/ejemplos_clase2.pdf> [Adcedida 20 mayo 2008].
- Craig Larman. *UML y Patrones: Introducción al análisis y diseño orientado a objetos*. Addison Wesley Longman.
- dzoom. *Formatos de almacenamiento de imagenes digitales* [Internet]. Disponible en: <http://www.dzoom.org.es/cont-38-formato-fichero-imagen-fotografia.html> [Adcedida 20 junio 2008].

- Darío de Miguel Benito y Juan José Pantrigo Fernández. *Detección automática del color de la piel en imágenes bidimencionales basado en el análisis de reguiones*. Universidad Rey Juan Carlos, España 2005. Disponible en: <http://www.escet.urjc.es/~jjpantrigo/PFCs/MemoriaPielFeb05.pdf> [Adcedida 20 junio 2008].

- *Statitical Color Models with Application to Skin Detection*. Dicember 1998, Cambridge Research Laboratory.

- Gopalan Suresh Raj. *The Factory Method Design Pattern* [Internet]. Disponible en: <http://gsraj.tripod.com/design/creational/factory/factory.html> [Adcedida 21 junio 2008].

Anexo A

Crecimiento de Internet

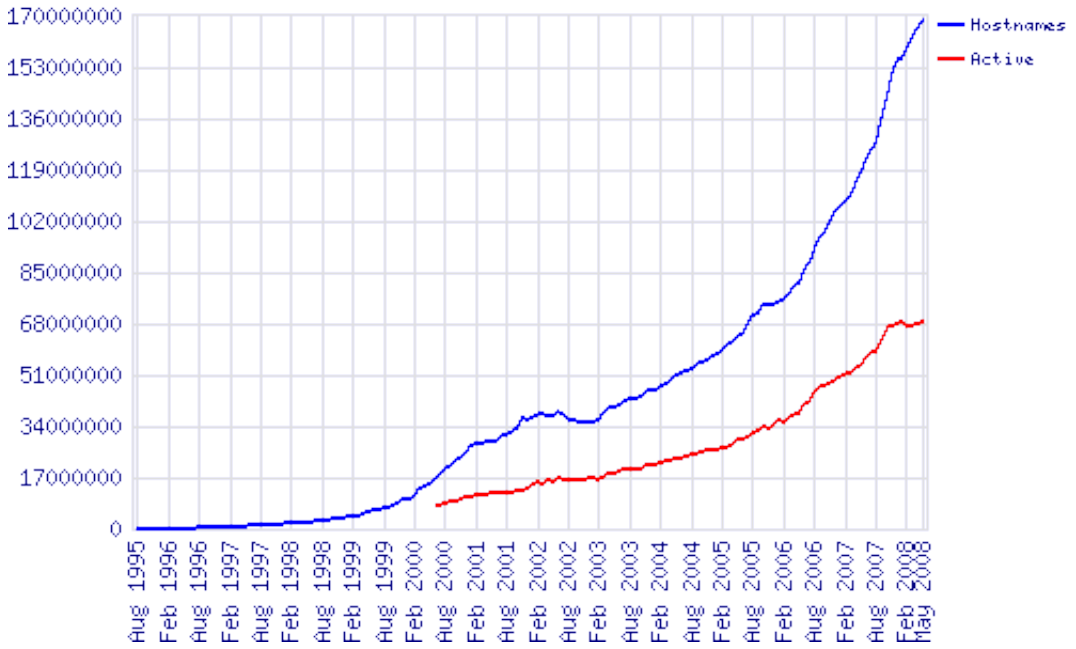

Figura A.1: Sitos Web a escala mundial (de 1995 a 2008).

Anexo B

Funcionamiento general de Filpacon

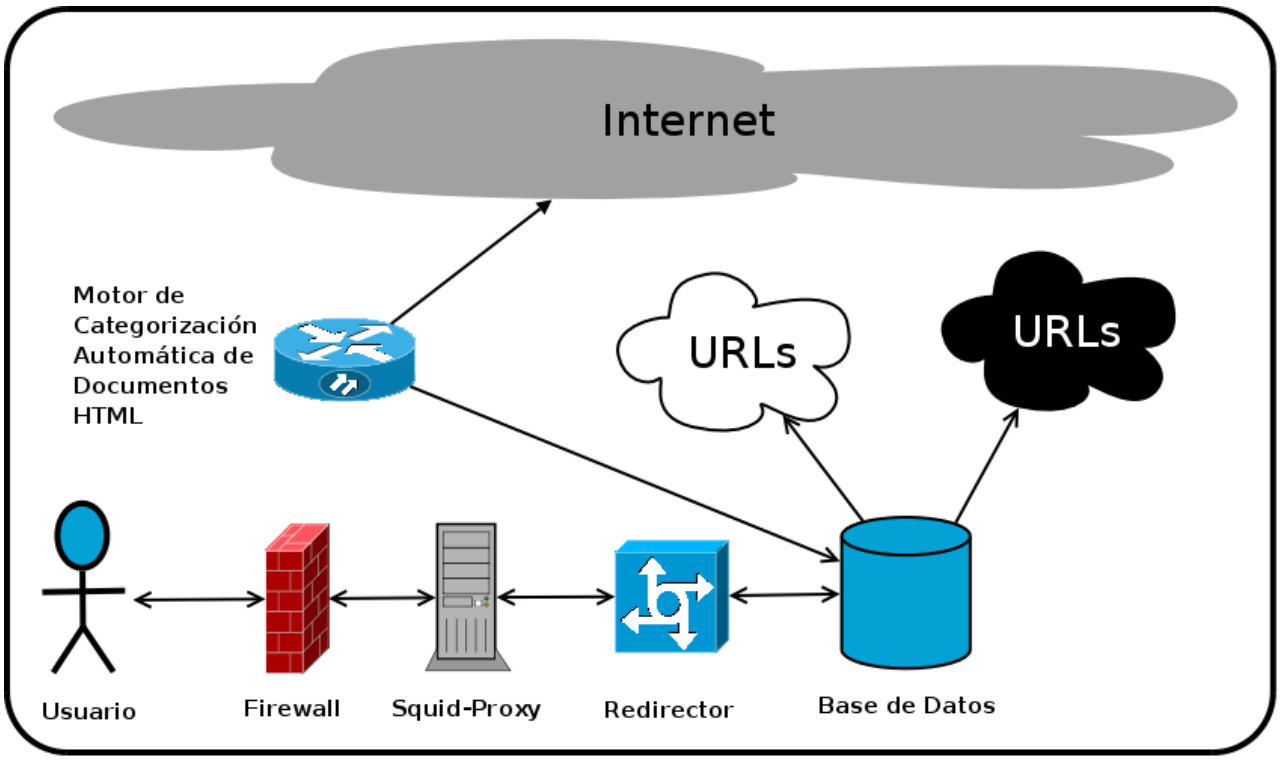

Figura B.1: Funcionamiento general de Filpacon

Anexo C

Perceptrón multicapa

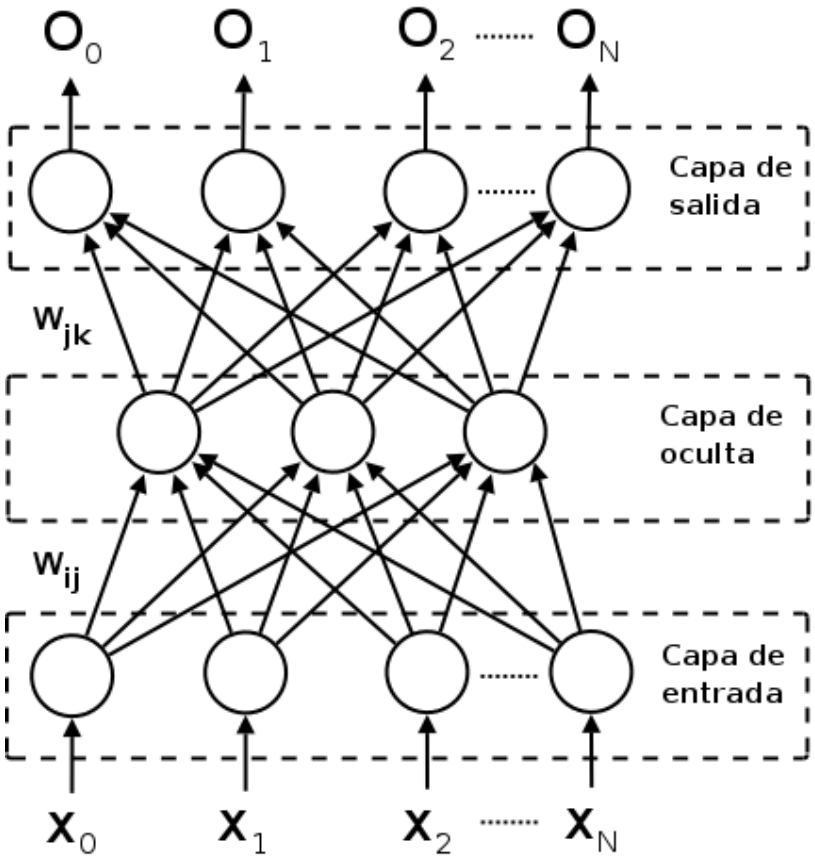

Figura C.1: Arquitectura de una Red Neuronal con propagación hacia atrás.

Anexo D

Fichero de configuración

```
# Run section
[run]
# The mode option can be: Train, Test or Prod
mode = Train # Change only the section of the mode,
             # for this mode change only train section [train]

# Train section
[train]
objectionable_imgs_path = ../data/training_data/objectionable_imgs/
bening_imgs_path = ../data/training_data/bening_imgs/

# Test section
[test]
objectionable_imgs_path = ../data/test_data/objectionable_imgs/
bening_imgs_path = ../data/test_data/bening_imgs/

# Prod section
[prod]
imgs_path = ../data/prod_data/

# Segementation section
```

```
[segmentation]
skin_histogram_path = ../data/skin-32-32-32.hist
non_skin_histogram_path = ../data/nonskin-32-32-32.hist
probability_map_conf = ../config/probability_map.conf
binary_img_umbral = 0.90
save_ima_mask = 0    # If you want save the image mask
save_binary_img = 0 # If you want save the binary image

# Classifier section
[classifier_mlp]
mlp_conf = ../config/mlp.conf
train_file_path = ../data/mlp_train.dat
objectionable_imgs_class_id = 1 # Don't change that
bening_imgs_class_id = 0        # Change if you realy sure what you do
```

Glosario de términos

UCI Universidad de las Ciencias Informáticas

FILPACON Filtrado de Paquetes por Contenido:
Es un sistema que pretende ser flexible y fiable para regular los contenidos nocivos de Internet.

IA Inteligencia Artificial:
Rama de la informática que desarrolla procesos que imitan a la inteligencia de los seres vivos.

MCADHTML Motor de Categorización Automatizada de Documentos HTML

OSRI Oficina de Seguridad para las Redes Informáticas

BDUC Base de Datos de URLs Categorizadas

HTML *HyperText Mark Language*:
Lenguaje de Marcas de Hipertexto. Es el lenguaje de marcado predominante para la construcción de páginas Web.

Web Red:
La traducción literal de esta palabra inglesa es tela de araña, pero en términos informáticos significa mucho más que eso.

SOFM *Self Organizing Feature Maps*:
Mapas de Características Autorganizados. Es una topología de Red Neuronal que se utiliza para la categorización no supervisada.

k-NN k *Nearest Neighbour*:
Proviene del idioma ingles y es traducido como los k vecinos más cercanos.

RNA Redes Neuronales Artificiales

UML *Unified Modeling Language*:
Lenguaje de modelado visual que se usa para especificar, visualizar, construir y documentar artefactos de un sistema de software.

POESIA Public Open-source Enviroment for a Safar Internet Access:
Es un sistema de filtrado de código abierto bajo la licencia GPL

OPTENET *Optimal Internet*:
Sistema de filtrado con más fuerza en Europa

URL Uniform Resource Locator:
Se refiere al texto que identifica a una página Web.

MCAIP Módulo de Categorización Automática de Imágenes Pornográficas

RUP *Rational Unified Process*:
Metodología de desarrollo de software creada por la Corporación Rational.

CASE *Computer Aided Sofware Engineering*:
Ingeniería de Software Asistida por Ordenador

OEM *Original Equipament Manufactured*:
Manufactura de Equipo Original. Empresa que compra un producto a un fabricante y lo integra en un producto propio.

ISP Internet Services Provider:
Proveedor de servicios de Internet. Una compañía que permite a sus usuarios tener acceso a Internet, normalmente a través de líneas telefónicas.

GNU *GNU is Not Unix*:
Sistema operativo cuyo nombre es un acrónimo recursivo que significa “GNU No es Unix”

Linux Es el núcleo (kernel) del sistema operativo libre GNU

GPL *General Public License*:
Licencia Pública General. Es una licencia creada por la Free Software Foundation (FSF) y orientada principalmente a los términos de distribución, modificación y uso del software.

MLP *Multilayer Perceptron*:
Perceptron multicapa. Topología de Red Neuronal definida por Rosenblatt en el año 1957.

IDE *Integrate Development Enviroment*:
Entorno de desarrollo integrado. Herramienta que se usa para facilitar el desarrollo de software.

CDT C/C++ *Development Tools*:
Es un pluging para Eclipse el cual permite la programación en el lenguaje C/C++

GDB GNU *Debugger*

ANSI *American National Standards Institute*:
Instituto de Estándares Nacional Americano.

STL *Standards Template Library*:
Biblioteca de Plantilla Estándar

IGU Interfaz Gráfica de Usuario:
Software que gestiona la interacción con el usuario de manera gráfica mediante el uso de iconos, menú, ratón, etc

CU Caso de Uso

JPEG *Joint Photographic Experts Group*:
Técnica de compresión de imágenes que casi no afecta la calidad y reduce hasta el 5 % del tamaño de la imagen.

PNG *Portable Network Graphics*:
Gráficos Portables de Red. Formato gráfico muy completo especialmente pensado para redes.

RGB *Red, Green and Blue*:
Rojo, verde y Azul. Es un modelo de color en el cual es posible representar un color mediante la mezcla de estos tres colores primarios.

HSV *Hue, Saturation and value*:
Modelo de color basado en tres parámetros tono, saturación y valor.

GRASP *General Responsibility Asignment Software Patterns*:
Patrones Generales de Software para Asignar Responsabilidades. Estos patrones describen los principios fundamentales de la asignación de responsabilidades a objetos, expresados en forma de patrones.

OODP *Object-Oriented Designe Patterns*:
Patrones de Diseño para la programación Orientada a Objeto.

WWW *World Wide Web*:
Telaraña o malla mundial. Sistema de información distribuido con mecanismos de hipertexto. Es el universo de servidores HTTP, que permiten mezclar texto, gráficos y archivos de sonido juntos.