ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Ingeniería Informática

Automatización de actividades para la enseñanza de la escritura occidental

Realizado por

Javier García-Herreros Castillero

Dirigido por

Beatriz Barros Blanco

Departamento de Lenguajes y Ciencias de la Computación

UNIVERSIDAD DE MÁLAGA

Málaga, Febrero 2009

UNIVERSIDAD DE MÁLAGA

ESCUELA TÉCNICA SUPERIOR DE INGENIERÍA INFORMÁTICA

Ingeniería Informática

Reunido el tribunal examinador en el día de la fecha, constituido por:

Presidente/a Dº/Dª. _____

Secretario/a Dº/Dª._____

Vocal Dº/Dª. _____

para juzgar el proyecto Fin de Carrera titulado:

Automatización de actividades para la enseñanza de la escritura occidental

del alumno/a Dº/Dª. Javier García-Herreros Castillero

dirigido por Dº/Dª. Beatriz Barros Blanco

ACORDÓ POR _____ OTORGAR LA CALIFICACIÓN DE _____

Y PARA QUE CONSTE, SE EXTIENDE FIRMADA POR LOS COMPARECIENTES DEL TRIBUNAL, LA PRESENTE DILIGENCIA.

Málaga, a de del 200_

El/La Presidente/a	El/La Secretario/a	El/La Vocal
Fdo:	Fdo:	Fdo:

Agradecimientos

A lo largo de estos años he recibido ayuda de mucha gente. A todos ellos les quiero agradecer el apoyo mostrado porque sin ese empujoncito, por mínimo que fuese, no habría podido llegar hasta aquí.

A título individual, quisiera empezar por la directora de este proyecto, la Dra. Beatriz Barros Blanco, que ha sabido en todo momento cómo sacar lo mejor de mí con la increíble dedicación y buen hacer que le caracteriza.

A mis compañeros Ale, Juan, Pedro y Rubén por esos días tan largos de estudio, de los cuales no podría haber salido vivo por mi cuenta (entre otras cosas porque jugar al Pro Evolution Soccer solo es un poco aburrido).

Y por supuesto a mi familia, por ser como son y haber estado ahí cuando les he necesitado.

Contenido

1. Introducción

Este proyecto se enmarca dentro de la educación infantil, en la fase del aprendizaje de la escritura. El aprendizaje del lenguaje escrito por parte de un niño no es un proceso rápido, sino que es complejo y prolongado, por lo que deberá pasar por una serie de etapas. En este proceso el niño deberá hacer un doble trabajo: por una parte deberá aprender los símbolos que componen el lenguaje y al mismo tiempo deberá acceder al lenguaje como un conjunto de diferentes géneros diferenciados (cuentos, noticias, instrucciones, etc). Según (Ferreiro & Teberosky, 1979), las fases del aprendizaje de la escritura se agrupan en preescritura y escritura, y se pueden describir del siguiente modo:

PREESCRITURA

-La primera fase en la que el niño diferencia la escritura del dibujo. En general, si el grafismo tiene una semejanza icónica con su referente se identifican como dibujo, y si no como escritura. En cambio, en esta fase los niños no consiguen realizar las letras convencionales. Es la etapa de la escritura diferenciada.

ESCRITURA

-En la segunda fase, a partir de la cual utilizan un repertorio de grafías convencionales: es la fase de la escritura diferenciada. El niño comienza a manejar diferentes hipótesis: linealidad, unión, discontinuidad, número mínimo de letras, etc.

-En la tercera, la fase silábica, los niños establecen relaciones entre los diferentes códigos y su aspecto sonoro, pero siempre regidos por la segmentación silábica de las palabras.

-La cuarta fase es la fase silábico-alfabética, donde los niños empiezan a tomar consciencia de la relación que existen entre las sílabas, pero todavía no son capaces de segmentar los elementos sonoros de la palabra.

-La quinta fase es la etapa alfabética, donde los niños conocen que a cada consonante y a cada vocal le corresponde una letra, aunque esto no signifique que su ortografía sea correcta.

Todas estas fases tienen asignados un conjunto de tipos de ejercicios que permiten al alumno practicar y adquirir destrezas. Generalmente se trata de unos patrones que los niños deben copiar de la forma más exacta posible. Es trabajo del educador el corregir cada uno de estos ejercicios que los alumnos han realizado en un papel y evaluar el trabajo. Dependiendo del resultado de esta evaluación, el educador deberá sopesar si el alumno debe hacer más hincapié en una parte determinada o puede pasar a ejercicios de más alto nivel. Parte de este proceso es informatizable,

sobre todo en lo referente a la gestión de los ejercicios y parte de la corrección de los mismos. Mediante tratamiento de imágenes se puede comparar el ejercicio del alumno y el patrón que debía seguir y de este modo realizar una calificación automatizada.

1.1. Estado del arte de programas para el aprendizaje de la escritura

Existen ya algunos sistemas que incorporan tecnología en el aula para niños (Nikolopoulou, 2007), tales como las herramientas derivadas del proyecto NIMIS (Lingnau, Hoppe, & Mannhaupt, 2003) que se centraban sobre todo en tareas de lecto-escritura y otras de carácter social. Por otro lado, también hay estudios como los de (Read J. C., 2007) que se dedican a analizar el nivel de usabilidad de este tipo de sistemas.

Este TFC se va a dedicar a la parte de escritura. Para el desarrollo correspondiente a la preescritura, hay otro trabajo paralelo que ha sido desarrollado por (de Diego, 2009) y dirigido por la misma profesora. En esa etapa también se utiliza un sistema de fases, al igual que en la escritura. Primero comienzan por líneas rectas, horizontales y verticales. A continuación continúan con líneas oblicuas, zigzag, círculos, combinaciones de ellos, etc. La principal diferencia entre dicho proyecto y este se encuentra en las diferentes características a evaluar, como puede ser el pulso del alumno, si dibuja la misma figura aunque esté un poco desplazada, el número de trazos que se ha utilizado para realizar el ejercicio, etc.

Otro programa relacionado con este PFC es el **abcteach** (www.abcteach.com), el cual es una herramienta que ayuda al profesorado a crear sus propios ejercicios/documentos cuando no encuentran los apropiados por otros medios. Estos documentos pueden ser de diversos tipos: crucigramas, sopa de letras, sudokus, listas de palabras para deletrear, ejercicios de ordenación alfabética de palabras, etc. Entre estos tipos se encuentra uno que realiza plantillas para el aprendizaje de la escritura. Tiene diferentes plantillas las cuales se diferencian en el tipo de letra, tamaño y estilo. A continuación se selecciona un tema a partir del cual se generan palabras relacionadas (si el tema es “invierno”, las palabras pueden ser nieve, frío…) y se crea un documento en formato PDF con el ejercicio. En principio no tenemos constancia de que pueda realizar una corrección automática ni que tenga ningún sistema adaptativo. Su utilización ser realiza mediante Web.

Figura 1: Captura de pantalla de abcteach

Existe un software llamado **sheets** el cual tiene plantillas de diferentes tipos de ejercicios. Estos están divididos en distintas categorías: actividades de comunicación donde los alumnos tienen que resolver los ejercicios hablando entre sí, actividades de vocabulario cuyo objetivo es que el alumno aprenda nuevas palabras y actividades de comprensión, utilizadas para fomentar el razonamiento de lo escuchado y sacar conclusiones. El sistema utilizado para crear los ejercicios es parecido al presentado en este PFC. Primero hay que elegir la plantilla deseada de las disponibles y a continuación hay que especificar el contenido. Una vez realizado este último paso, el programa da la opción al usuario de que pueda configurar y editar el ejercicio insertando objetos, modificando el aspecto final, etc. Por último se puede guardar el ejercicio e imprimirlo.

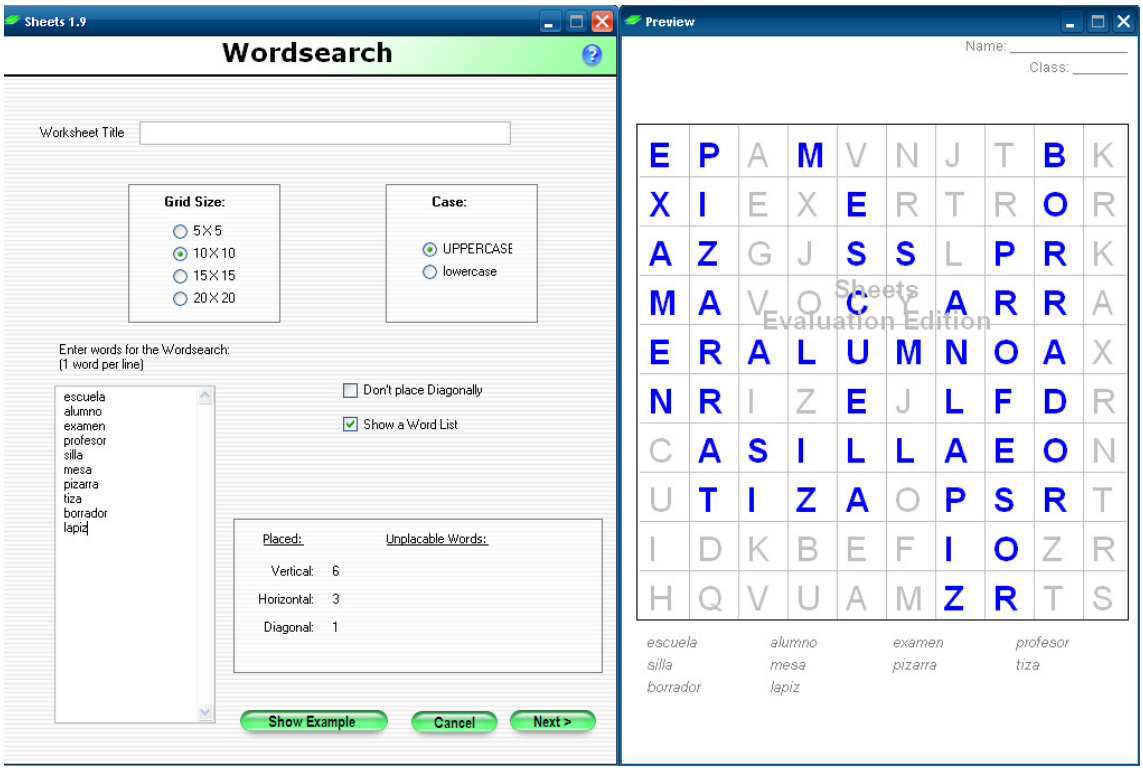

Figura 2: Captura de pantalla de Sheets

ComPET (Computerized Penmanship Evaluation Tool) es un software desarrollado por la Dra. Sara Rosenblum para diagnosticar problemas en la escritura, ya que pueden dar pistas de otro tipo de desórdenes y/o enfermedades como pueden ser problemas neuronales, de comportamiento, ADHD (Síndrome de déficit de Atención con Hiperactividad), párkinson, etc. Este sistema consiste en un papel y lápiz digital que detecta la presión y que están conectados a una tableta digital. Mediante este sistema los niños deben copiar una historia corta para evaluar aspectos como el tiempo de escritura, el tiempo que está el lápiz en el aire, velocidad, aceleración, espaciado y presión. Este proceso completo dura unos cinco minutos. Los niños con problemas de escritura suelen utilizar más movimientos para escribir una misma letra y tienden a dudar más a la hora de realizar los trazos.

Starwrite (www.startwrite.com) proporciona un conjunto de herramientas para crear ejercicios de escritura. Además de proveer un conjunto amplio de fuentes adecuadas para la enseñanza de la caligrafía, aporta métodos para poder insertar imágenes en los ejercicios, ya sea de las que tenga el usuario o las que vienen prediseñadas. Se pueden crear dibujos básicos mediante las herramientas de edición (círculos, cuadrados, líneas, etc), imprimir los ejercicios directamente, etc. En resumen, es un procesador de texto centrado en los niños, con herramientas específicas que hacen el diseño de los ejercicios mucho más fácil.

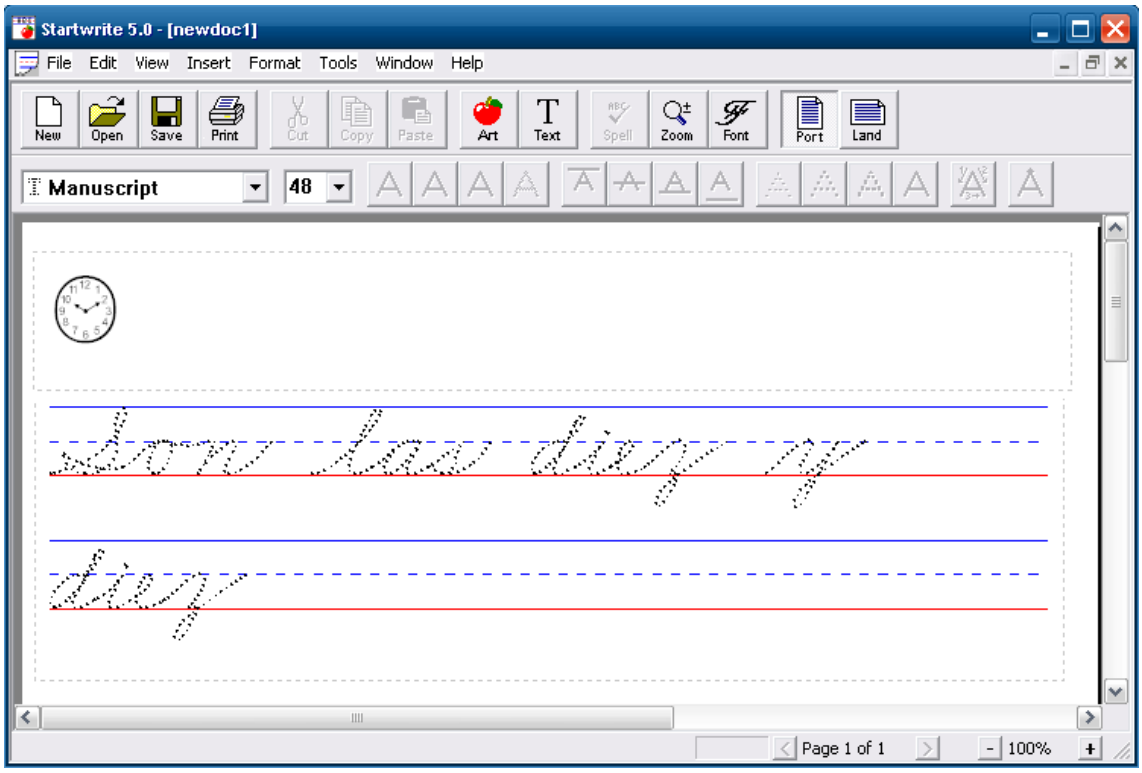

Figura 3: Captura de pantalla de startwrite

En la Tabla 1 se puede ver una comparación de las diferentes herramientas.

Característica	abcteach	sheets	ComPET	Startwrite
Plantillas	S	S	N	N
Generación automática de ejercicios	N	N	-	N
Evaluación automática	N	N	S	N
Utilización de varios dispositivos de entrada	N	N	N	N
Centrado en la escritura de los niños	N	N	S	S
Posibilidad de definir nuevos ejercicios	S	S	-	S
Adaptativo	N	N	N	N
Modelado de usuario	N	N	N	N
Creación de PDF's	S	N	-	N

Tabla 1: Resumen de características de los distintos programas

La mayoría de estos programas carecen de un sistema de generación automática de ejercicios ya que están enfocados a ayudar al alumno en un intervalo corto de tiempo, o bien a detectar niños con dificultades, etc. Sin embargo nuestro proyecto va más allá. Pretende ayudar al alumno y al profesor durante toda la fase de aprendizaje del alumno, es decir, desde que se entra en colegio a los 3 años hasta que el alumno adquiera una habilidad suficiente como para tener una caligrafía correcta, que pueden ser 8, 9, 10 años… Lo que considere oportuno el educador.

1.2. Objetivos

El objetivo del proyecto es automatizar parte del proceso de asignación y gestión de las tareas de escritura (a partir de la segunda fase, es decir, posterior a la pre-escritura) y apoyar la labor del docente, que ahora podrá monitorizar la evolución de cada alumno de manera detallada y además tener almacenados digitalmente los ejercicios resueltos, ordenados de forma cronológica para poder estudiar el proceso de evolución del aprendizaje. Este objetivo se desglosa en los siguientes subobjetivos:

Crear una herramienta que permita que el profesor defina ejercicios con diferentes niveles de dificultad, utilizando plantillas de tareas (Avgeriou, Papasalouros, Retalis, & Skordalakis, 2003).

Gestión de usuarios mediante perfiles. Cada usuario se autenticará con un nombre de usuario y contraseña y dispondrá de un perfil que represente su nivel de conocimiento actual. Esta estructura proporcionará información de cuánto sabe el alumno según diferentes parámetros evaluables así como posibles disgrafías o problemas que puedan inferirse a partir de éstos.

Desarrollar un sistema de evaluación automática, que tomará un ejercicio realizado por el alumno, generará un informe detallado tanto del nivel de conocimiento como de disgrafías y actualizará el perfil correspondiente.

Proporcionar un mecanismo que permita la administración de ejercicios de forma adaptativa teniendo en cuenta aquellos realizados por el alumno, su evaluación y necesidades de aprendizaje.

Implementación de diferentes interfaces para la realización de los ejercicios por parte de los alumnos. Se podrán evaluar ejercicios impresos y posteriormente escaneados, realizados directamente en un tabletPC o sobre la tableta.

Incorporar el concepto de trayectoria al perfil del alumno para poder representar y monitorizar su proceso de aprendizaje a lo largo de un periodo de tiempo.

1.3. Método de trabajo

Para el desarrollo del proyecto se va a seguir una metodología de diseño iterativo (Wilson, Rauch, & Paige, 1992) y evaluación formativa (Scriven, 1991). Esta forma de desarrollo consiste en ir realizando una serie de prototipos que van a ir siendo evaluados por el profesor y el alumno, refinando en cada iteración del modelado los requisitos del software. En este caso, se ha pensado realizar al menos tres iteraciones:

1.- En la primera fase se han de estudiar las letras que se deberán utilizar en la realización de los ejercicios. Se deberán proporcionar suficientes fuentes para satisfacer los requisitos de los diferentes niveles de los ejercicios. En unos casos se necesitará una fuente con cuadrícula, en otro simplemente unas líneas que guiarán al alumno para escribir correctamente. También se deberá poder cambiar el tamaño de la letra, y cualquier parámetro que pueda influir en la dificultad del ejercicio.

Además se ha de proporcionar suficiente flexibilidad en cuanto a la configuración del papel (márgenes, tamaño de papel, etc) para que el educador configure la página de la forma que considere óptima según sus criterios.

Se insertarán las plantillas de formatos, que se utilizarán para definir cuántas líneas de patrón y cuántas líneas a rellenar por el alumno habrá y sus posiciones relativas.

Crear una interfaz de usuario intuitiva para la realización de los ejercicios, plantilla, etc.

El prototipo resultante de esta iteración deberá poder crear los ejercicios manualmente (es decir, configurados por el educador) para su posterior evaluación.

2.- En el segundo ciclo se modelarán las diferentes fases del aprendizaje de la escritura que se utilizan hoy en día. El hecho de estar en una determinada fase determinará la longitud de las palabras que se deben utilizar en los ejercicios, la dificultad de unir las diferentes letras (hay determinadas uniones más complicadas que otras), restringirá el tamaño de la letra (en fases más avanzadas la letra será más pequeña), el margen de error que el alumno puede cometer, etc.

Se realizará un modelo de usuario con el que se podrán realizar todas las modificaciones necesarias para que el educador evalúe el trabajo de los alumnos. Es decir, desde diseñar los ejercicios de forma automática o manual, asignarlos a los alumnos, corregirlos, realizar resúmenes, consultar los ejercicios resueltos...

Se implementará un sistema de corrección automática que evaluará diferentes características del ejercicio realizado por el alumno.

Se proporcionará diferentes métodos de entrada para la corrección/realización de ejercicios. Por un lado se podrán imprimir, escanear y evaluar el ejercicio a partir de la imagen escaneada. Por otro lado se podrá utilizar un tabletPC con pantalla táctil para que los alumnos realicen el ejercicio directamente sobre él. También se podrá usar un tablet que guarda el trazo realizado por un alumno en una tarjeta SD y a partir de ahí evaluarlo.

Se deben crear diferentes perfiles de usuario para restringir el uso del programa. El método de autenticación del usuario se realizará a través de una pantalla de login donde se requerirá un nombre de usuario y una palabra de paso. Cada usuario deberá estar asociado a un perfil.

3.- En la tercera iteración se mejorará el sistema incluyendo una parte de monitorización de los alumnos.

Además se deberá crear un modelo de creación automática de ejercicios que se basará en los ejercicios realizados anteriormente por los alumnos y sus resultados. De esta forma, si un alumno falla en un determinado ejercicio, automáticamente se creará uno parecido para que pueda corregir los errores cometidos con anterioridad.

Opcionalmente se le podrá incluir un módulo de avisos automáticos para que el educador siempre esté informado del progreso de cada uno de los alumnos.

1.4. Organización de la memoria

La memoria consta de cuatro capítulos. El capítulo actual es el primero, que pretende servir de introducción al usuario al problema que se va a abordar. Además se analizan algunos de los programas que existen en la actualidad que tienen objetivos comunes con este trabajo. Se explica el proceso que se va a seguir para la realización del mismo y los objetivos que se pretenden conseguir.

En el segundo capítulo se repasa la información previa que el lector va a necesitar para la perfecta comprensión del trabajo. En primera instancia se habla de los diferentes sistemas y/o dispositivos utilizados para obtención de datos por parte del sistema, como pueden ser un tabletPC o un escáner. A continuación se expone el concepto de metadato, para qué se utilizan, cómo se usan, etc. También se explican los sistemas adaptativos, y qué utilidad tienen. Las plantillas y formatos también tienen su espacio, ya que son la base para el diseño de los ejercicios para los alumnos. Seguidamente se explica el código de barras que se ha utilizado para la identificación de los ejercicios escaneados (Code 128). Se aborda el tema de la corrección de los ejercicios por parte de los docentes, las características que se tienen en cuenta para realizar la evaluación, etc. Por último se habla del tratamiento de imágenes, diferentes tipos de filtros, su uso, etc.

El tercer capítulo es el centro del trabajo ya que trata del diseño del sistema, la toma de datos, el modelo de usuario… es decir, se documenta el funcionamiento del sistema. Además se muestra la interfaz de usuario y cómo utilizarlo para poder sacarle todo el provecho.

En cuarto lugar se explica cómo se han implementado las funcionalidades más importantes del sistema, como es la base de datos, la interfaz, etc. Se hace un ligero repaso sobre las tecnologías utilizadas al comienzo del capítulo.

A continuación, en el quinto capítulo, se expone el tipo de desarrollo de software que se ha seguido para el desarrollo del proyecto y los diferentes ciclos de implementación, evaluación, etc.

Por último se encuentra un capítulo sobre las conclusiones y los futuros trabajos que se le pueden realizar al sistema.

2. Elementos para el diseño de un sistema de apoyo a la enseñanza de la escritura

Como se ha comentado en el punto 1.4, este apartado pretende explicar al lector los diferentes conceptos que va a necesitar a lo largo de la lectura de esta memoria para su óptima comprensión, ya que todos ellos han sido utilizados en alguna parte del sistema.

2.1. Dispositivos para el aprendizaje de la escritura

En este apartado se van a describir un conjunto de dispositivos que se pueden utilizar como herramientas de apoyo en el aula que amplían el abanico de posibilidades para que el niño practique y aprenda las destrezas necesarias para escribir bien en un futuro próximo. El objetivo que se persigue con el uso de estos aparatos es, por un lado, formativo con intención de observar cuáles de ellos realmente ayudan al aprendizaje y cuáles no, y por otro, tecnológico, para observar qué mecanismo de entrada de datos favorece mejor el desarrollo de las destrezas motrices del aprendiz. A lo largo de esta sección se listarán un conjunto de dispositivos que van relacionados a una forma de introducción de datos que van desde el papel, la más tradicional hasta el tabletPC donde todo el proceso de realización de ejercicios de escritura está soportado por ordenador.

2.1.1. Papel y escáner

El primer sistema que puede utilizarse para el aprendizaje es la conjunción del papel junto con un escáner para la evaluación automatizada mediante el ordenador. La forma de utilización es la creación de un ejercicio, el cual se imprimirá en una hoja de papel y se entregará al alumno para su realización.

La ventaja de este sistema reside en el mínimo impacto que tiene para el alumno, el cual ya está familiarizado con el uso del papel y el lápiz. De esta forma sólo deberá preocuparse de su escritura sin que ningún otro factor requiera su atención. Con otro tipo de dispositivos el alumno deberá aprender su manejo, con lo que los primeros ejercicios realizados no reflejarán su capacidad de escritura sino que también influirá en el resultado la destreza que haya adquirido en el uso del dispositivo.

2.1.2. Pantalla táctil de un tabletPC

Un tabletPC es un ordenador/PDA con pantalla táctil, donde se puede escribir sobre la misma con un lápiz que lleva integrado. Estos dispositivos pueden parecerse más a un ordenador portátil, con teclado, ratón, etc, a los que se le puede rotar la pantalla y utilizarlos en forma de "libreta", o bien pueden carecer de estos dispositivos y acercarse más a una PDA pero con unas dimensiones mayores.

La ventaja de estos dispositivos para el aprendizaje de la escritura es su parecido a la escritura en papel. Además los ejercicios se corrigen automáticamente justo al terminar de realizarlos, el siguiente está disponible inmediatamente (no hace falta imprimirlo), etc.

Tiene dos desventajas claras respecto a la utilización del papel y el escáner: el plano donde se muestran las imágenes está bastante por debajo del plano donde se apoya el lápiz, de tal forma que si no se mira con suficiente perpendicularidad, el punto presionado puede no dibujarse justo en la punta del lápiz, con la consiguiente desorientación para los niños. Este problema se puede solucionar calibrando la pantalla, pero en cuanto se cambie el ángulo de visión, se volverá a tener el mismo problema. La otra desventaja es que este tipo de pantallas tiene un retardo mínimo desde que se pasa el lápiz por una zona hasta que se dibuja, por lo que también tiende a confundir a los alumnos si escriben demasiado rápido.

Figura 4: Utilización de tabletPC

2.1.3. Tableta gráfica

La tableta gráfica es un dispositivo de entrada de datos consistente en una tableta que captura la presión ejercida por un lapicero especial que viene incluido con ésta. De esta forma se pueden dibujar gráficos a mano, escribir, etc. Existen dos tipos

de tabletas, las que se conectan a un ordenador como periférico y las que son independientes. Éstas últimas suelen tener alguna forma de almacenamiento donde guardar lo que se ha escrito/dibujado. Generalmente suelen ser tarjetas de memoria SD y/o similares.

Para el aprendizaje de la escritura es preferible el segundo tipo. A este tipo de tabletas se le coloca el ejercicio en papel sobre ellas y el alumno lo realiza normalmente. La única diferencia es que mientras el alumno realiza el ejercicio sobre el papel, se guarda en la memoria la realización del ejercicio en tiempo real.

Las ventajas que tiene este método es que es más natural para el alumno escribir sobre un papel, como han venido haciendo desde edades más tempranas.

El principal inconveniente reside en que el papel debe estar fijo para que la captura sea consistente con lo que el alumno ha escrito. Si el papel se mueve algo (lo cual puede ser totalmente normal, ya que el alumno suele apoyarse, cambiar de posición, etc.) con respecto a la tableta, el resultado obtenido no va a capturar con exactitud lo escrito por el alumno y por tanto la evaluación no va a ser correcta.

Figura 5: Tableta gráfica Genius

2.2. Metadatos

Una de las claves del sistema automático de generación de ejercicios son los metadatos. Estos se utilizan para clasificar, describir e identificar, los diferentes datos del sistema, como las plantillas, contenido de los ejercicios, cada paso del workflow,

etc. De esta forma, para cada etapa del aprendizaje se conoce con qué elementos se puede construir un ejercicio.

La palabra “metadato” significa literalmente “sobre el dato”, aunque la acepción más genérica es “dato sobre los datos”. Otra clase de definiciones trata de precisar el término como “descripciones estructuradas y opcionales que están disponibles de forma pública para ayudar a localizar objetos” o “datos estructurados y codificadas que describen características de instancias conteniendo informaciones para ayudar a identificar, descubrir, valorar y administrar las instancias descritas”. Esta clase de definiciones hace mayor hincapié en los metadatos en relación con la **recuperación de información**, y surgió de la crítica de que las declaraciones más simples son tan difusas y generales que dificultarán la tarea de acordarse de estándares, pero estas definiciones no son muy comunes.

Los metadatos pueden describir colecciones de objetos y también los procesos en los que están involucrados, describiendo cada uno de los eventos, sus componentes y cada una de las restricciones que se les aplican. Los metadatos definen las relaciones entre los objetos, como las tuplas en una base de datos o clases en orientación a objetos, generando estructuras.

El primer uso que se les conoce fue la catalogación de libros en las bibliotecas, donde se generaba una descripción del contenido de los libros.

Los metadatos tienen tres funciones básicas:

- Proporcionar una descripción de una entidad de información junto con otra información necesaria para su manejo y preservación.
- Proporcionar los puntos de acceso a esa descripción.
- Codificar esa descripción.

Un **registro de metadatos** consiste en un conjunto de atributos, o elementos necesarios para describir la fuente en cuestión. En nuestro caso el conjunto de metadatos de un paso del workflow puede ser por ejemplo la dificultad que se requiere en esa etapa, el tipo de letra, el tamaño de la misma, etc. Esta relación entre el metadato y el dato al que describe se puede dar de dos formas:

- Los elementos pueden estar en un registro separado del documento, como en nuestro caso, donde existe una tabla en la BD que relaciona los pasos del workflow con sus metadatos, por ejemplo.
- Los metadatos pueden estar incluidos, incrustados en el propio recurso.

Ejemplos de uso de los metadatos pueden ser:

- El encabezamiento de un fichero multimedia (imagen, vídeo o audio).
- El resumen de un documento.
- El catálogo de una base de datos.
- Los términos asignados haciendo uso de un tesauro.
- Las palabras extraídas de un texto.
- Las fichas catalográficas en cualquier formato (ISBD, MARC, etc.).
- Las páginas amarillas.

Los **beneficios derivados de la utilización de metadatos** son diversos y dependen del área en que se utilicen.

En términos generales:

- Los metadatos **adhieren contenido, contexto y estructura a los objetos de información**, asistiendo de esta forma al proceso de recuperación de conocimiento desde colecciones de objetos.
- Los metadatos **permiten generar distintos puntos de vista conceptuales** para sus usuarios o sistemas, y liberan a estos últimos de tener conocimientos avanzados sobre la existencia o características del objeto que describen.
- Los metadatos **permiten el intercambio de la información** sin la necesidad de que implique el intercambio de los propios recursos.
- En cada proceso productivo, o en cada etapa del ciclo de vida de un objeto de información, se van generando metadatos para describirlos y metadatos para describir dichos metadatos (manual o automáticamente) generando de esta forma **valor añadido** a los recursos
- Los metadatos permiten un **acceso a los recursos en forma controlada** ya que se conoce con precisión el objeto descrito.
- Los metadatos permiten **preservar los objetos de información** permitiendo migrar (gracias a la información estructural) sucesivamente éstos, para su posible uso por parte de las futuras generaciones.

Los metadatos son esenciales para sostener un crecimiento de una Web a mayor escala, permitiendo **búsquedas, integración y recuperación del conocimiento** desde un mayor número de fuentes heterogéneas.

2.3.Plantillas y formatos

Según (Wikipedia, 2009), una **plantilla** se puede definir como "*una forma de dispositivo que proporciona una separación entre la forma o estructura y el contenido.*

Es un medio o un instrumento que permite guiar, portar o construir un diseño o esquema predefinido.

Una plantilla agiliza el trabajo de reproducción de muchas copias idénticas o casi idénticas (que no tiene que ser tan elaborado, sofisticado o personal). Si se quiere un trabajo más refinado, más creativo, la plantilla no es sino un punto de partida, un ejemplo, una idea aproximada de lo que se quiere hacer, o partes comunes de una diversidad de copias."

En nuestro caso las plantillas van a resultar útiles a la hora de crear los ejercicios, de tal forma que la estructura se mantenga inalterada pero se pueda cambiar el contenido, es decir, el patrón que tenga que copiar el alumno. Este método es conveniente debido a que la plantilla es la que define la dificultad del ejercicio, por lo que normalmente se reutilizará para los diferentes alumnos, pero debido a que cada alumno tiene sus propias necesidades, deberemos cambiar el contenido para adaptarnos mejor a cada uno de ellos. También son útiles cuando queremos que el alumno realice diferentes ejercicios pero con el mismo nivel de dificultad.

En nuestro caso los **formatos** son la distribución de los diferentes elementos que se tiene el ejercicio. Un ejercicio consta de tres de ellos: **líneas de patrón o texto**, que serán las que el alumno deba copiar, **líneas en blanco** que será donde el alumno deba copiar el patrón y **dibujos** para hacer el ejercicio más atractivo y estimulante. La única restricción que existe es que antes de una línea en blanco debe haber una línea de patrón, ya que de otro modo no habría nada que copiar en la línea en blanco.

Estos formatos son independientes de cada plantilla. La única relación entre ellos es que cada plantilla está asociada a un formato, es decir, que sólo tendrá una disposición posible de sus elementos. Pero a una plantilla se le puede asignar cualquier formato y un formato puede estar asociado a multitud de plantillas (o a ninguna).

2.4. Sistemas adaptativos

Un sistema adaptativo es aquel que es capaz de comportarse de diferente forma dependiendo del usuario que esté interaccionando con él. En nuestro caso, dependiendo del alumno el sistema debe crear unos ejercicios u otros, dejar acceder a determinadas partes del sistema, etc. Una de las partes más importantes de los sistemas adaptativos es el modelo de usuario. El modelo de usuario es la representación de información sobre un usuario concreto para que el sistema pueda adaptarse a las necesidades del mismo. Por ejemplo, si un usuario realiza una búsqueda de determinada información, el sistema puede seleccionar y/o priorizar los elementos más relevantes para dicho usuario dependiendo de la información guardada anteriormente. Para crear y mantener el modelo de usuario actualizado, el sistema obtiene datos de diferentes fuentes que puede incluir la observación del comportamiento del usuario con el sistema, requerir al usuario que inserte determinados datos, etc. Este proceso se conoce como modelado de usuario. La

cantidad de datos recogida y su naturaleza dependen directamente de la magnitud de la adaptación que se le requiera al sistema.

De acuerdo con la naturaleza de la información que se modela en los sistemas adaptativos, se pueden distinguir modelos que representan características del usuario como individuo o bien modelos que representan el contexto del trabajo actual del usuario. Nosotros nos centraremos en las cinco características más útiles y populares cuando se utilizan el primer tipo de modelos:

- El conocimiento del usuario
- Intereses
- Metas
- Antecedentes
- Información personal

El conocimiento del usuario es la característica más importante en los sistemas adaptativos educacionales. La forma más simple de un modelo del conocimiento del usuario es un modelo escalar que estima el nivel del mismo con un solo valor, ya sea cuantitativo (por ejemplo un número en el rango [0-100]) o cualitativo (por ejemplo, bueno, medio, bajo, etc). A pesar de su simplicidad este tipo de modelo puede ser muy efectivo para técnicas de adaptación no demasiado complejas. El problema de el modelo escalar es su baja precisión. En un dominio de tamaño razonable, el conocimiento del usuario puede ser bastante diferente para distintas partes del dominio. Por ejemplo, en procesamiento de texto, un usuario puede ser un experto en anotaciones de texto pero un novato en edición de fórmulas. Un modelo escalar efectivo debe guardar una media del conocimiento del usuario en el dominio completo. Para técnicas de adaptación avanzadas se requiere un modelo más complejo que el escalar. Por estas razones los sistemas adaptativos educacionales que se centran en un conocimiento del usuario complejo utilizan varios tipos de modelos estructurales. Estos modelos estructurales suponen que el dominio del conocimiento se puede dividir en fragmentos independientes e intentan representar el conocimiento del usuario en cada uno de estos fragmentos. Dependiendo de la naturaleza del conocimiento representado, los modelos estructurales pueden ser clasificados independientemente en dos sub-dimensiones:

- El tipo del conocimiento representado
- Una comparación del conocimiento del usuario con el nivel del conocimiento de un experto, es decir, la meta a la que se quiere llegar.

La forma más popular del modelo de conocimiento estructural es el modelo superpuesto. El propósito de este modelo es representar el conocimiento del usuario como un subconjunto del modelo del dominio, el cual refleja el nivel del sujeto. Para

cada fragmento del dominio, el modelo superpuesto guarda la estimación del nivel de conocimiento del usuario de ese fragmento. El modelo superpuesto puro asigna a cada fragmento un valor lógico, verdadero o falso, indicando si el usuario conoce o no ese fragmento. En un modelo superpuesto moderno, el valor representa el grado de conocimiento del fragmento. Puede ser un valor cualitativo o cuantitativo.

Puesto que el modelo superpuesto representa el conocimiento del usuario como un subconjunto del conocimiento ideal, la naturaleza de dicho conocimiento reflejada en este modelo depende directamente de la naturaleza del conjunto que se quiere representar, es decir, del conocimiento ideal. La mayoría de los sistemas tutorizados inteligentes (ITS) se centran en representar dos tipos de dominio del conocimiento: conocimiento conceptual (hechos y sus relaciones entre sí) y conocimiento de procedimental (habilidad de resolver problemas). El conocimiento conceptual típicamente se representa mediante una red de conceptos. El conocimiento de procedimental generalmente se representa como un conjunto de reglas de resolución de problemas. Hay muchos tipos de representación del conocimiento que permiten combinar estos dos tipos.

Los modelos superpuestos constituyen un paso adelante con respecto a los modelos escalares. Aún así, en el campo de los ITS, estos modelos a menudo han sido criticados por ser demasiado simples ya que el conocimiento del usuario a veces no es un subconjunto del conocimiento ideal. El usuario puede tener errores de comprensión y generalmente para aumentar su conocimiento, el camino no es "aprender lo que no se sabe" sino refinar y modificar el conocimiento existente. Para modelar los errores de compresión del usuario, el modelo superpuesto se mejoró a un modelo de errores, donde se representa tanto el conocimiento correcto como el incorrecto.

Los intereses del usuario han sido uno de los datos más importantes (y a veces el único) a tener en cuenta a la hora de diseñar un sistema adaptativo. Sin embargo, los nuevos sistemas educacionales adaptativos tienen más en cuenta las metas de aprendizaje que los intereses del usuario.

La meta de los usuarios o tareas representan dentro de un sistema adaptativo el propósito del trabajo del usuario. Dependiendo del tipo de sistema, puede ser la meta del trabajo (en sistema de aplicaciones), información que se requiera en ese momento (en sistemas de acceso a la información) o una meta de aprendizaje (en sistemas educacionales, como es nuestro caso). En todos estos casos la meta es la respuesta a la pregunta "¿Qué quiere conseguir el usuario?". La meta del usuario es una de las características más cambiantes. Casi siempre cambia de sesión en sesión, incluso puede cambiar varias veces dentro de la misma sesión.

El modelado de metas en los sistemas educacionales adaptativos modernos se consigue mediante un catálogo de metas, el cual es parecido al modelado del conocimiento superpuesto. La base de este modelado es un catálogo predefinido de posibles metas o tareas que el sistema pueda reconocer. Muchas veces este catálogo es simplemente un conjunto pequeño de metas independientes, sin embargo, algunos sistemas utilizan un catálogo más avanzado en forma de una jerarquía de tareas. En este modelo, metas genéricas de alto nivel se van descomponiendo a su vez en sub-metas, hasta llegar al nivel más bajo, las hojas del árbol, que serían metas a corto plazo. Típicamente el sistema asume que el usuario tiene una única meta en un momento determinado. El trabajo de adaptación consiste en averiguar cuál es esa meta y marcarla como la meta actual en el modelo. En nuestro caso, dependiendo de la meta del alumno en un momento dado se le generará un tipo de ejercicio u otro (con una fuente más grande o pequeña, con más letras de un determinado tipo, etc.).

Los antecedentes del usuario es un nombre común que se le da al conjunto de características relacionada con la experiencia que tiene el usuario fuera del dominio específico del sistema. Esto puede ser la profesión del usuario, experiencia de trabajo en determinadas áreas, etc. Por ejemplo, un sistema adaptativo médico puede cambiar la información mostrada o la terminología médica dependiendo del tipo de usuario (médico, enfermero, etc.).

La naturaleza de los antecedentes es similar a la naturaleza del conocimiento del usuario, sin embargo la representación y el mantenimiento es diferente ya que los antecedentes no son tan importantes como el conocimiento. Debido a esto, el modelado de los antecedentes es mucho más simple, como puede ser un modelo estereotipado. También hay que tener en cuenta que los antecedentes raramente cambian en un usuario, por lo que generalmente estos datos se introducen explícitamente y no se deducen de la interacción con el sistema, como es el caso del conocimiento.

2.5. Método de escritura

El flujo de trabajo del sistema se puede resumir en el esquema de la Figura 6 en el que puede verse los diferentes elementos generados por todos los actores implicados.

Se diferencian tres zonas:

- Verde: El responsable de realizar las operaciones en esta zona es el educador
- Roja: El alumno es el encargado de ejecutar las acciones.
- Azul: El sistema se encargará del procesado necesario.

Primero se debe generar la plantilla de los diferentes ejercicios, bien de forma manual, bien automáticamente. Para este segundo caso se le han de asociar unos metadatos. Dependiendo de estos datos, el generador automático de ejercicios dispondrá de información suficiente sobre qué plantilla elegir.

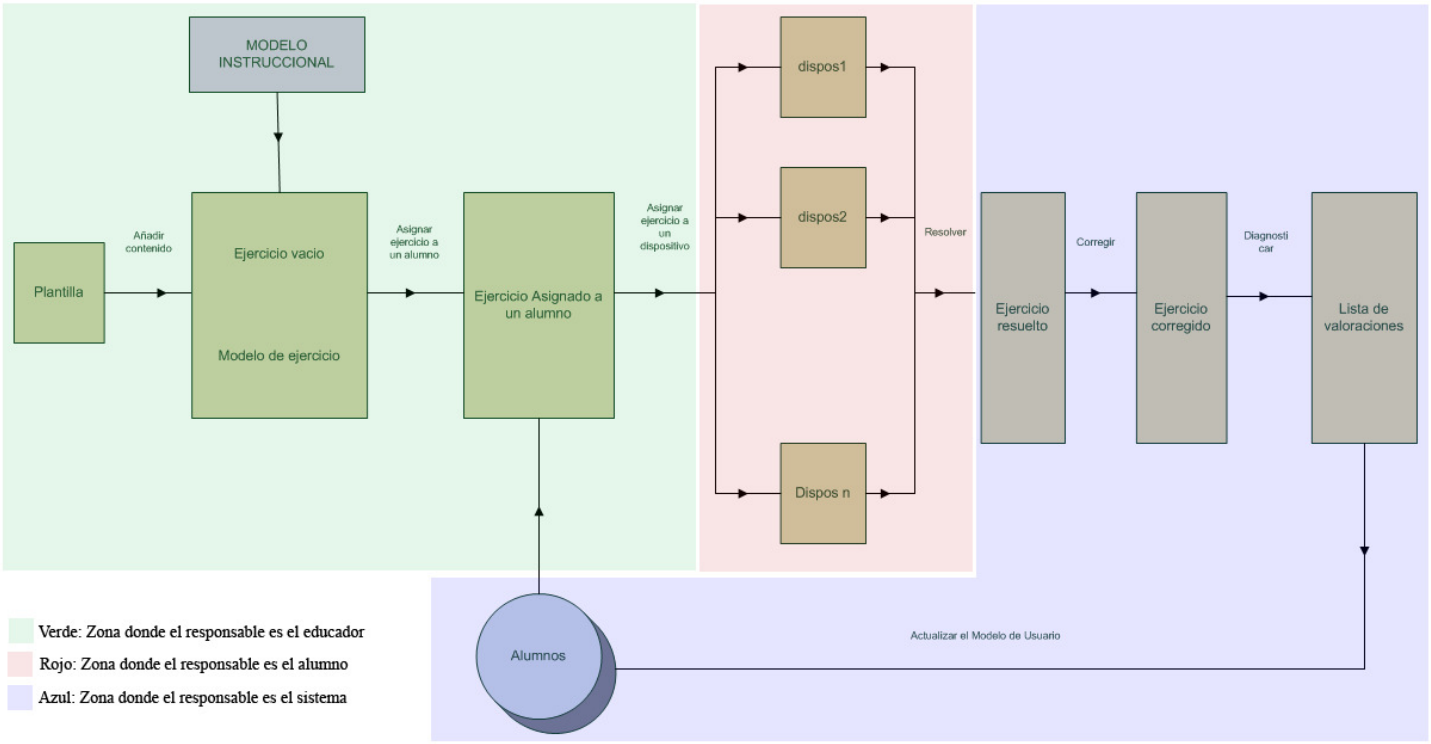

Figura 6: Esquema de workflow

Una vez generada la plantilla, se han de introducir los contenidos teniendo en cuenta el modelo instruccional que se lleve a cabo en cada colegio. Estos contenidos se pueden basar en patrones o se pueden introducir manualmente. El problema con la introducción manual es que hay que generar demasiados ejercicios para que exista una variedad suficiente para asegurar que hay uno para cada necesidad. Sin embargo, mediante la construcción con patrones se pueden generar los que se necesiten en cada momento, por ejemplo un ejercicio con muchas 'm'.

A continuación, estos ejercicios creados se asignan a un alumno, el cual los podrá resolver con diferentes dispositivos, como un tabletPC o con papel (y el posterior escaneo del mismo). Este proceso puede ser manual o automático. En general, cuando un alumno debe hacer ejercicios primero realizará aquellos que se le hayan seleccionado de forma manual para dar prioridad al docente. Si el alumno no tiene ejercicios disponibles, se le asignará uno nuevo teniendo en cuenta sus necesidades de aprendizaje. Si existen ejercicios que se adecuen a estas y todavía no han sido realizados por el alumno, se elegirá uno entre los posibles. Si no, se generará uno atendiendo a su modelo de usuario.

Seguidamente, el alumno deberá resolver los ejercicios que tenga disponibles mediante el dispositivo de entrada seleccionado (papel y escáner o tabletPC).

Una vez obtenido el ejercicio resuelto, se corrige mediante una serie de algoritmos y se obtiene una evaluación del mismo. Al interpretar estos datos, se actualiza el modelo de usuario correspondiente, es decir, qué ha realizado, sus resultados, etc. Dependiendo de estas evaluaciones se generarán nuevos ejercicios, adecuando su nivel al del niño y haciendo hincapié en aquellos aspectos donde dé muestras de un nivel bajo.

2.6. Identificación de usuarios con códigos de barras

A la hora de corregir los ejercicios hay que conocer cuál es, qué alumno lo ha realizado, etc. Si se utiliza el tabletPC no hay problema, ya que se conoce el alumno (bien porque se haya identificado en el sistema o bien porque se haya seleccionado directamente), y también se sabe qué ejercicio se está mostrando en cada momento. En este caso se tiene toda la información disponible para hacer la corrección, pero cuando se realiza en papel y se escanea, estos datos no se pueden obtener directamente.

Para ello se ha diseñado un sistema de código de barras. A la hora de imprimir el ejercicio, se crea este código, que indica el nombre del alumno, su ID (identificador) y el ID del ejercicio. De esta forma se puede corregir el ejercicio sin que se requiera la inserción de datos por parte de la persona que esté realizando los escaneos.

El sistema de código de barras que se ha utilizado ha sido “Code 128” que puede codificar todos los caracteres ASCII de 7 bits (128 caracteres). Se basa en un sistema con 4 anchos diferentes para las barras y los espacios para conseguir el código de barras más compacto posible. La única restricción en la longitud del código de barras que puede existir es la del tamaño físico que pueda tener, que no se ajuste a las especificaciones del producto o de la maquinaria usada para imprimir.

Existen tres subconjuntos del “Code 128”:

- CODE 128A: Codifica los caracteres ASCII desde el 0 hasta el 95, lo que incluye los números del 0 al 9, las letras mayúsculas (sin contar la ñ), caracteres de control y especiales.
- CODE 128B: Codifica los caracteres ASCII desde el 32 hasta el 127, es decir, letras mayúsculas, minúsculas y caracteres especiales.
- CODE 128C: Codifica pares de números, desde el 00 hasta el 99.

Al principio del código de barras existe un carácter de control para indicar con qué subconjunto se comienza. Cada subconjunto también codifica caracteres propios

de control para indicar el cambio de subconjunto en medio del código de barras, entre otros motivos.

La estructura de un código de barras Code 128 tiene los siguientes elementos:

- Una zona en blanco a la izquierda del código de barras
- El carácter de comienzo
- Los caracteres de datos
- Un dígito de control
- El carácter de finalización
- Una zona en blanco a la derecha del código de barras

Las zonas en blanco deben ser al menos diez veces más anchas que la línea o espacio en blanco más estrecho.

Cada carácter se compone de tres barras y tres espacios en blanco, excepto el carácter de finalización, que contiene una barra más al final. Como habíamos dicho anteriormente cada barra/espacio en blanco puede tener 4 anchos diferentes, que son múltiplos del más pequeño. Es decir, si la barra más estrecha tiene 3 píxeles, los demás anchos serán 6, 9 y 12. No puede existir un ancho de 5, por ejemplo.

Los códigos que definen los diferentes caracteres se pueden ver en la Tabla 2.

Value	Code Set A	Code Set B	Code Set C	Bar/Space Pattern B S B S B S	Value	Code Set A	Code Set B	Code Set C	Bar/Space Pattern B S B S B S
0	SP	SP	00	2 1 2 2 2 2	**54**	V	V	54	3 1 1 1 2 3
1	!	!	01	2 2 2 1 2 2	**55**	W	W	55	3 1 1 3 2 1
2	"	"	02	2 2 2 2 2 1	**56**	X	X	56	3 3 1 1 2 1
3	#	#	03	1 2 1 2 2 3	**57**	Y	Y	57	3 1 2 1 1 3
4	$	$	04	1 2 1 3 2 2	**58**	Z	Z	58	3 1 2 3 1 1
5	%	%	05	1 3 1 2 2 2	**59**	[	[	59	3 3 2 1 1 1
6	&	&	06	1 2 2 2 1 3	**60**	\	\	60	3 1 4 1 1 1
7	'	'	07	1 2 2 3 1 2	**61**	]	]	61	2 2 1 4 1 1
8	(	(	08	1 3 2 2 1 2	**62**	^	^	62	4 3 1 1 1 1
9	)	)	09	2 2 1 2 1 3	**63**	_	_	63	1 1 1 2 2 4
10	*	*	10	2 2 1 3 1 2	**64**	NUL	`	64	1 1 1 4 2 2
11	+	+	11	2 3 1 2 1 2	**65**	SOH	a	65	1 2 1 1 2 4
12	,	,	12	1 1 2 2 3 2	**66**	STX	b	66	1 2 1 4 2 1
13	-	-	13	1 2 2 1 3 2	**67**	ETX	c	67	1 4 1 1 2 2
14	.	.	14	1 2 2 2 3 1	**68**	EOT	d	68	1 4 1 2 2 1
15	/	/	15	1 1 3 2 2 2	**69**	ENQ	e	69	1 1 2 2 1 4

16	0	0	16	1 2 3 1 2 2	**70**	ACK	f	70	1 1 2 4 1 2
17	1	1	17	1 2 3 2 2 1	**71**	BEL	g	71	1 2 2 1 1 4
18	2	2	18	2 2 3 2 1 1	**72**	BS	h	72	1 2 2 4 1 1
19	3	3	19	2 2 1 1 3 2	**73**	HT	i	73	1 4 2 1 1 2
20	4	4	20	2 2 1 2 3 1	**74**	LF	j	74	1 4 2 2 1 1
21	5	5	21	2 1 3 2 1 2	**75**	VT	k	75	2 4 1 2 1 1
22	6	6	22	2 2 3 1 1 2	**76**	FF	l	76	2 2 1 1 1 4
23	7	7	23	3 1 2 1 3 1	**77**	CR	m	77	4 1 3 1 1 1
24	8	8	24	3 1 1 2 2 2	**78**	SO	n	78	2 4 1 1 1 2
25	9	9	25	3 2 1 1 2 2	**79**	SI	o	79	1 3 4 1 1 1
26	:	:	26	3 2 1 2 2 1	**80**	DLE	p	80	1 1 1 2 4 2
27	;	;	27	3 1 2 2 1 2	**81**	DC1	q	81	1 2 1 1 4 2
28	<	<	28	3 2 2 1 1 2	**82**	DC2	r	82	1 2 1 2 4 1
29	=	=	29	3 2 2 2 1 1	**83**	DC3	s	83	1 1 4 2 1 2
30	>	>	30	2 1 2 1 2 3	**84**	DC4	t	84	1 2 4 1 1 2
31	?	?	31	2 1 2 3 2 1	**85**	NAK	u	85	1 2 4 2 1 1
32	@	@	32	2 3 2 1 2 1	**86**	SYN	v	86	4 1 1 2 1 2
33	A	A	33	1 1 1 3 2 3	**87**	ETB	w	87	4 2 1 1 1 2
34	B	B	34	1 3 1 1 2 3	**88**	CAN	x	88	4 2 1 2 1 1
35	C	C	35	1 3 1 3 2 1	**89**	EM	y	89	2 1 2 1 4 1
36	D	D	36	1 1 2 3 1 3	**90**	SUB	z	90	2 1 4 1 2 1
37	E	E	37	1 3 2 1 1 3	**91**	ESC	{	91	4 1 2 1 2 1
38	F	F	38	1 3 2 3 1 1	**92**	FS	|	92	1 1 1 1 4 3
39	G	G	39	2 1 1 3 1 3	**93**	GS	}	93	1 1 1 3 4 1
40	H	H	40	2 3 1 1 1 3	**94**	RS	~	94	1 3 1 1 4 1
41	I	I	41	2 3 1 3 1 1	**95**	US	DEL	95	1 1 4 1 1 3
42	J	J	42	1 1 2 1 3 3	**96**	FNC 3	FNC 3	96	1 1 4 3 1 1
43	K	K	43	1 1 2 3 3 1	**97**	FNC 2	FNC 2	97	4 1 1 1 1 3
44	L	L	44	1 3 2 1 3 1	**98**	SHIFT	SHIFT	98	4 1 1 3 1 1
45	M	M	45	1 1 3 1 2 3	**99**	CODE C	CODE C	99	1 1 3 1 4 1
46	N	N	46	1 1 3 3 2 1	**100**	CODE B	FNC 4	CODE B	1 1 4 1 3 1
47	O	O	47	1 3 3 1 2 1	**101**	FNC 4	CODE A	CODE A	3 1 1 1 4 1
48	P	P	48	3 1 3 1 2 1	**102**	FNC 1	FNC 1	FNC 1	4 1 1 1 3 1
49	Q	Q	49	2 1 1 3 3 1	**103**	Start A	Start A	Start A	2 1 1 4 1 2
50	R	R	50	2 3 1 1 3 1	**104**	Start B	Start B	Start B	2 1 1 2 1 4
51	S	S	51	2 1 3 1 1 3	**105**	Start C	Start C	Start C	2 1 1 2 3 2
52	T	T	52	2 1 3 3 1 1	**106**	Stop	Stop	Stop	2 3 3 1 1 1 2

53	U	U	53	213131					

Tabla 2: Caracteres Code 128

El dígito de control es un checksum módulo 103. Se calcula sumando el valor del carácter de comienzo a los productos de la posición de los caracteres por su valor. El carácter más a la izquierda tiene posición 1. El siguiente, 2, etc. El resultado se divide por 103 y el resto es el valor del carácter de control. Por ejemplo, para calcular el carácter de control de la cadena 'Code 128':

```
                          Valor                Total
                          =====                =====
     Código de comienzo B   104                  104
Posición 1              C    35        1 x 35 =   35
Posición 2              o    79        2 x 79 =  158
Posición 3              d    68        3 x 68 =  204
Posición 4              e    69        4 x 69 =  276
Posición 5                    0        5 x  0 =    0
Posición 6              1    17        6 x 17 =  102
Posición 7              2    18        7 x 18 =  126
Posición 8              8    24        8 x 24 =  192
                                                 =====
                                                 1197
                                                 =====
            1197/103 = 11 con resto 64
```

Figura 7: Ejemplo de comprobación de checksum

El valor del carácter de control en el ejemplo anterior sería 64, que corresponde según la tabla al carácter ' ` ' (código ASCII 96).

2.7. Indicadores para la corrección de ejercicios

Una vez que los alumnos han superado la fase de preescritura y han aprendido los movimientos básicos de la escritura, el objetivo es mejorar la calidad de esta, es decir, su caligrafía. Para ello existen determinados aspectos que hay que vigilar y corregir en caso necesario. Los indicadores a tener en cuenta son los siguientes:

Claridad

En este apartado lo que se pretende es que la forma de las letras sea la correcta, sin deformaciones ni distorsiones:

- Control de los ganchos de las letras que los llevan (l, t,...), mantener los ángulos bucles,...etc.
- Cuidar los enlaces entre las grafías, de forma que no se produzcan amontonamientos entre las letras.
- No comprimir ni abombar los elementos circulares, etc. Evitar abolladuras.

Uniformidad

Proporción entre las grafías, regularidad entre las diferentes alturas, etc. Para ello hay que conocer la proporción de las letras respecto a la pauta. Existen cuatro tipos de letras:

- La que sobresale por encima y por debajo: f
- Las que sólo sobresalen por encima: b, d, h, l, k, t
- Las que sólo sobresalen por debajo: g, j, p, q, y
- Las que no sobresalen de la pauta y, por tanto, son del mismo tamaño: a, c, e, i, m, n, ñ, o, r, s, u, v, w, x, z

Para cuidar esta proporción se recomienda que los niños comiencen a escribir con la doble pauta, es decir, líneas que especifiquen las diferentes alturas de las letras. Habrá dos líneas que remarquen la pauta central y será la parte superior e inferior de letras que no sobresalen, como la a, la c, etc. Además existirán otras dos líneas exteriores que marcarán la altura de las letras que sobresalen por encima y por debajo.

Espaciamiento

Separación correcta entre los distintos elementos gráficos de la escritura:

- Separación entre las letras para evitar soldaduras y amontonamientos
- Separación entre las palabras
- Separación entre línea y línea

Ligaduras

Es el aspecto relacionado con los enlaces entre las letras. Para favorecer un buen espaciamiento y unas buenas conexiones entre las letras el procedimiento didáctico más adecuado es el de realizar el aprendizaje gráfico en cursiva instruyendo claramente los enlaces entre las letras.

Inclinación

Es importante mantener siempre la misma inclinación en la escritura. Los tres tipos posibles de inclinación son: dextrógiros (hacia la derecha), levógiros (hacia la izquierda) y vertical.

La inclinación más recomendable para mantener un buen espaciamiento, una buena proporción y una buena uniformidad entre grafías será la dextrógira.

Presión-rapidez

Estos dos factores están directamente relacionados ya que una presión excesiva impedirá una rapidez adecuada y viceversa. Una velocidad elevada hará que la presión sea inferior.

Una presión normal es aquella que evita que aparezcan situaciones no recomendables como tensión, temblores, calambres, molestias en la mano, etc. Esto a su vez podría producir paradas bruscas, arrancadas impulsivas, etc., por lo que no habrá un desplazamiento normal del brazo/mano.

Tareas específicas que facilita una buena relación entre presión y rapidez son las de los grandes trazos enlazados, especialmente bucles en todas las modalidades expuestas.

Disposición de la página

Habrá que cuidar los márgenes, la limpieza, el alineamiento entre líneas, etc.

2.8. Tratamiento de imágenes

Uno de los dispositivos de entrada al sistema es el escáner, el cual convierte el ejercicio del alumno en una imagen digital. Esta imagen necesita ser tratada mediante diferentes técnicas para extraer la información requerida para poder evaluar automáticamente un ejercicio. En este apartado vamos a realizar un breve repaso a cada una de ellas para poder entender los procedimientos que se siguen en la segmentación.

2.8.1. Introducción a las imágenes digitales y su tratamiento

Una imagen digital no es más que una matriz de números enteros. En general cada uno de los colores está representado por 8 bits [0-255]. Si una imagen sólo tiene un color (escala de grises) cada pixel ocupará esos 8 bits. Sin embargo las imágenes más comunes son a color, por lo que cada pixel estará representado por 24 bits debido a que cualquier color se puede representar como una combinación de tres colores: Rojo, verde y azul (RGB en sus siglas en inglés). Existen también imágenes que además tienen otra coordenada, *alpha*, que representa la transparencia de ese pixel, pero por ahora no la vamos a tenerla en cuenta. Por tanto una imagen a color se puede representar de dos formas: como una matriz donde cada elemento ocupa 24 bits o bien como tres matrices (una por color) donde cada elemento ocupa 8 bits.

El problema es que no sólo hay que interpretar esos números sino que en el proceso de digitalización se ha podido introducir ruido y/o efectos espurios que pueden hacernos malinterpretar esos números. Por tanto, lo primero que hay que hacer es tratar esa imagen para eliminar estos posibles errores que pueda haber. Esto se realiza con un filtrado espacial. Además, dependiendo de la luminosidad, el tipo de digitalización, etc., el contraste de la imagen puede ser muy pobre así como la luminosidad.

2.8.2. Conceptos básicos en procesamiento de imagen

Histograma

Un histograma de una imagen es una representación gráfica de la frecuencia con la que aparecen los diferentes colores de la imagen. De una imagen en escala de grises sólo se podrá obtener un histograma, el cual tendrá en la coordenada X los valores del 0 al 255 (correspondiendo con los diferentes tonos de grises que puede tomar la imagen) y en eje de ordenadas la frecuencia con la que aparecen dichos tonos. Para una imagen a color existirá un histograma por cada color primario (rojo, verde y azul). Adicionalmente se puede mostrar un único histograma para una imagen a color que represente la luminosidad de la imagen. Para ello hay que convertir la imagen a escala de grises y calcularlo. Como el ojo humano percibe diferente luminosidad dependiendo del color que se esté viendo no se puede realizar una simple media de los tres colores. La ecuación de la luminancia que representa ese fenómeno teniendo en cuenta la sensibilidad del ojo humano a cada una de las frecuencias del espectro visible es la siguiente:

$$Y = 0.3R + 0.59G + 0.11B$$

Estos histogramas se construyen recorriendo toda la imagen y contabilizando el número de píxeles que poseen cada nivel. De esta forma podemos obtener información estadística de cómo están distribuidos los diferentes niveles de gris o de color. Esto es útil para saber si la digitalización se ha efectuado correctamente.

Por ejemplo, en el caso de fotografías submarinas, debido a que el agua absorbe en primera instancia la longitud de onda correspondiente al color rojo, nos podemos encontrar fotografías con problemas de color como los que tiene la de la Figura 8:

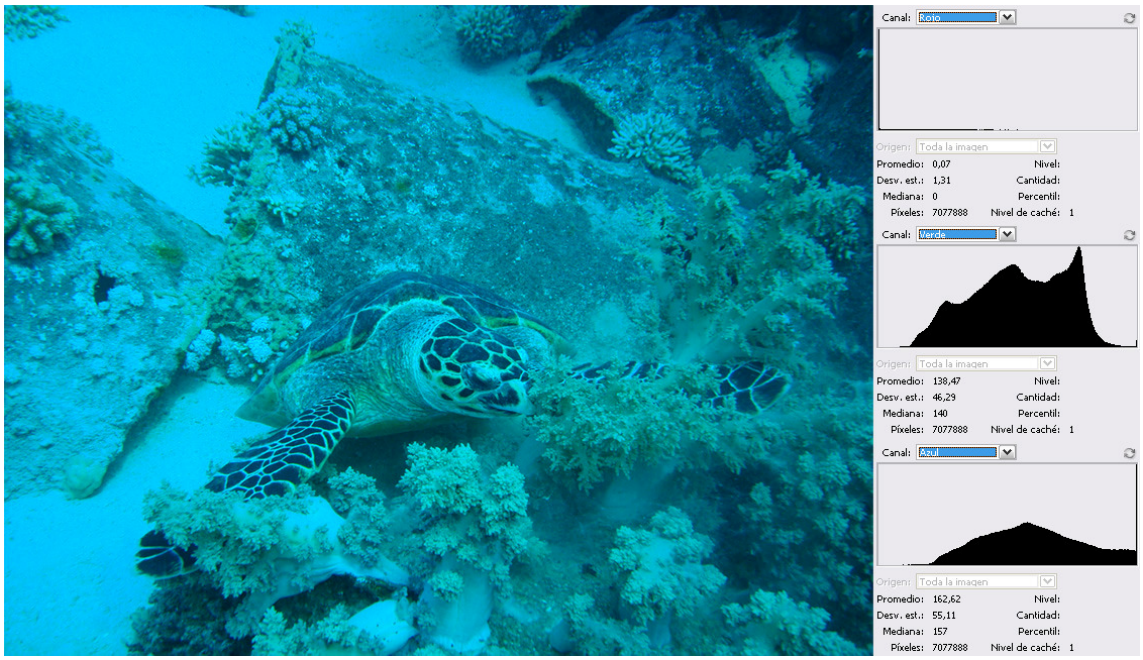

Figura 8: Fotografía submarina con una deficiente digitalización

Dependiendo de la cámara, ajustará el balance de blancos de una forma u otra. Pero como se puede comprobar en los histogramas y a simple vista, el color rojo es prácticamente nulo (el correspondiente al primer histograma). Esta información nos servirá para guiarnos en el retoque de la fotografía y darle unos colores más naturales, como se puede observar en la siguiente figura:

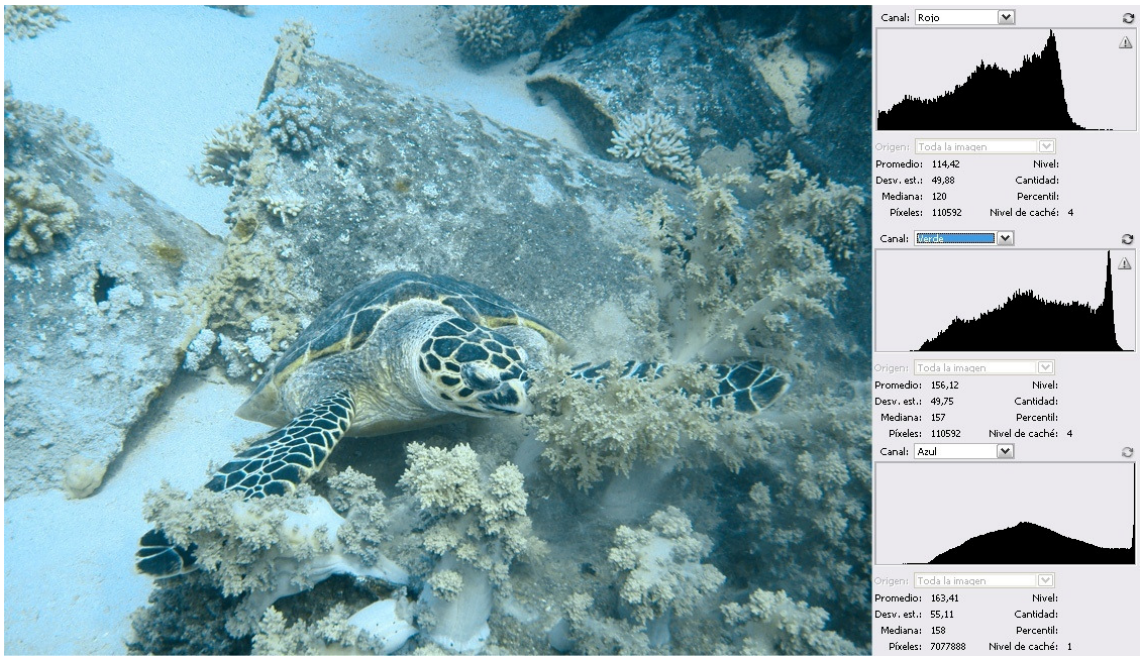

Figura 9: Fotografía submarina con histogramas corregidos

Lo que se ha realizado en esta fotografía ha sido una modificación en los tonos de verde (cuyo histograma era bastante amplio) pasándolos a rojo mediante una

función de transformación que se puede observar en la Figura 10. Como se puede comprobar en los histogramas ahora los tonos verdes han dejado paso a tonos de rojo, dando de esta forma unos colores mucho más naturales.

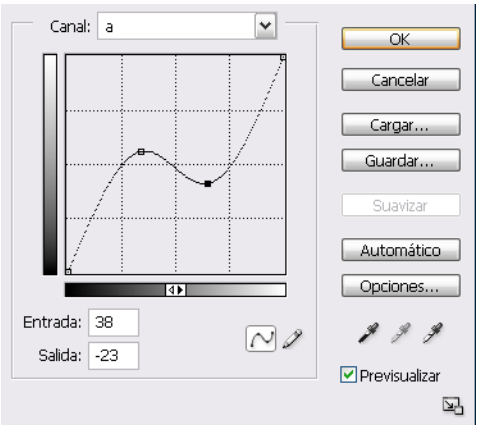

Figura 10: Curva corrección color verde a magenta

En nuestro caso, a partir de los histogramas de una imagen podemos aproximar qué valores son los que toman las letras que ha realizado el alumno ya que sabemos que el fondo es blanco, por lo que todos los colores que se aproximen a esa zona se pueden descartar. De esta forma se puede ajustar la curva de corrección para poder obtener los datos que nos interesan de forma óptima.

Contraste

El contraste de una imagen es un parámetro que nos indica las diferencias de intensidad entre los píxeles que la componen. Será alto si la diferencia es grande y será bajo si la imagen es muy homogénea. ¿Cómo se puede ver el contraste de la imagen en un histograma? Pues dado que las imágenes homogéneas tienen bajo contraste, ya que todos sus píxeles son muy parecidos, el histograma estará concentrado en una zona determinada de niveles de gris. Esto hace que, al apreciar poco los cambios de color, perdamos información. Este hecho se puede mejorar en cierta medida con funciones de transformación, las cuales "estiran" el histograma para que los distintos valores se repartan a lo largo de toda la paleta y se aprecien mejor los cambios.

Una medida del contraste se puede obtener mediante la varianza de la intensidad de los píxeles que componen la imagen. Si todos los píxeles son iguales, la varianza será cero y cuanto mayor sea la diferencia entre ellos, mayor será la varianza.

Para la corrección de los ejercicios escaneados es muy importante este parámetro, ya que tendremos que modificarlo de tal forma que se aumente la diferencia de valores entre lo escrito por el alumno y la cuadrícula del fondo.

Brillo

El brillo de una imagen es la media de los valores de los píxeles. Mientras más alta sea esta media, más alto será el brillo. Se puede observar que el brillo es básicamente un sinónimo de luminosidad. Los colores claros, que son los que tienen el valor más alto, se perciben como más luminosos que los colores oscuros, con valores más bajos.

Para modificar el brillo sin modificar el contraste basta con desplazar el histograma hacia un lado o hacia otro. Si se desplaza hacia valores más bajos, el brillo disminuirá. Si por el contrario se desplaza hacia valores más alto, el brillo aumentará. Dado que la varianza de los píxeles es la misma ya que los valores no han cambiado con respecto a la media, el contraste se mantiene inalterado.

Tabla de consulta

Una tabla de consulta es una función que modifica los distintos niveles de los colores/tonos de gris. Dado que los niveles de cada color o de los tonos de gris se encuentran en el rango [0-255], tanto el dominio de la función como el rango será dicho conjunto.

Se pueden considerar distintos tipos de tablas de consulta. Pueden ser un conjunto de pares (*valor de entrada*, *valor de salida*), funciones continuas, lineales, no lineales, lineales a tramos, etc.

La función que devuelve la imagen sin modificar es $y = x$, donde por cada valor de entrada x se devuelve el mismo valor de salida, y. Cabe destacar que esta función tiene como pendiente 1. Si en una determinada zona, la función tiene una pendiente mayor podemos concluir que en ese intervalo se está aumentando el contraste ya que un intervalo de un determinado tamaño va a corresponder a un intervalo mayor de salida, es decir, el intervalo de salida va a tener mayor diferencia entre sus píxeles que el intervalo de entrada. Un ejemplo de este tipo de función es el siguiente:

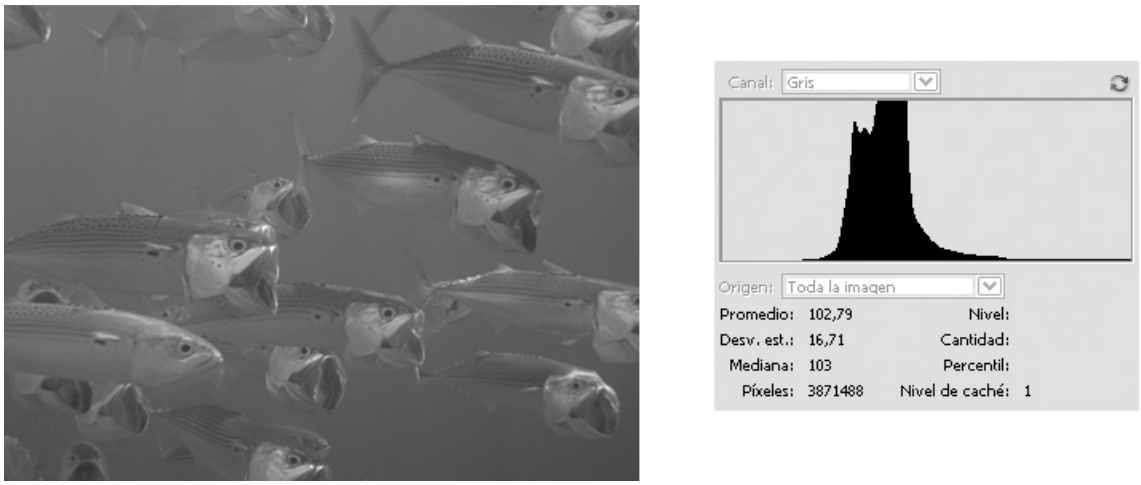

Figura 11: Imagen original con poco contraste

Esta es la imagen original. Como se puede observar tiene muy poco contraste ya que los tonos de gris están concentrados en una sola zona del histograma. Si aplicamos la siguiente función que aparece en la Figura 12, obtenemos la imagen mostrada en la Figura 13.

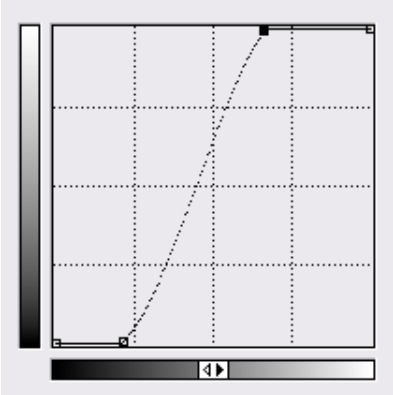

Figura 12: Función de cambio de contraste

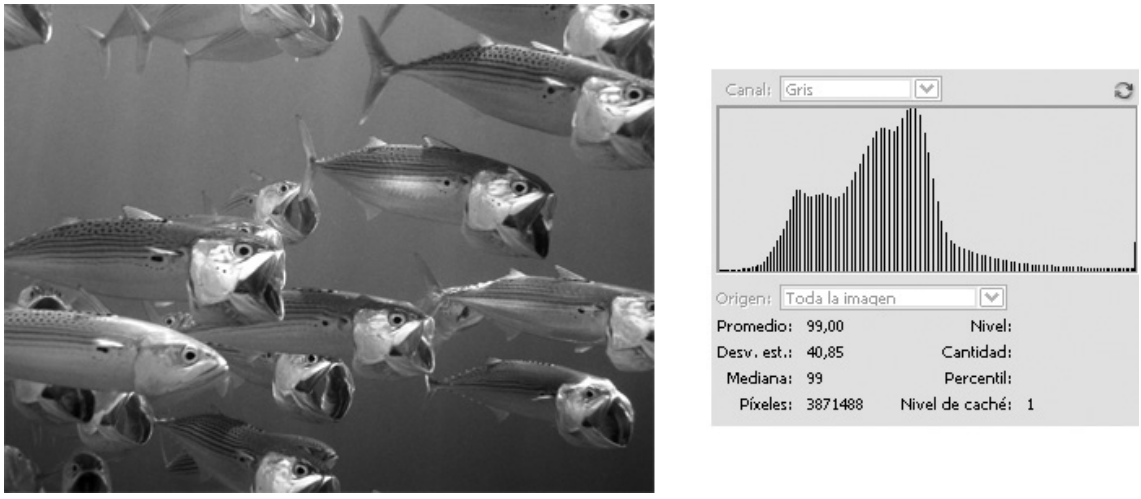

Figura 13: Imagen con el contraste aumentado

Como se puede observar la curva tiene una pendiente mayor que 1 en aquella zona donde el histograma estaba concentrado. De esta forma conseguimos "estirar" el histograma para que se puedan apreciar mejor los cambios de color. Hay que darse cuenta, como dijimos anteriormente, que la desviación estándar del histograma ha subido de 16.71 (contraste muy pobre) a un valor de 40.82.

Una elemento a tener en cuenta al realizar estas operaciones es mantener un equilibrio entre la ganancia de contraste y la pérdida de información que se pueda dar en los extremos, es decir, si aumentamos más el contraste (estiramos el histograma) hay valores que, siendo diferentes toman el valor máximo al aplicar la función, con lo que se crearía una saturación de blanco, que ya se ha empezado a dar en la imagen de muestra. Se puede ver la pequeña línea vertical en 255 que indica un número alto de píxeles de ese color.

Este tipo de función se utilizará para cambiar el contraste y el brillo de los ejercicios que se escaneen para poder obtener con mayor facilidad lo escrito por el alumno.

2.8.3. Filtros espaciales

Los filtros espaciales tienen como finalidad modificar la contribución de determinados rangos de frecuencias a la formación de la imagen. Un cambio de gris muy brusco en la imagen necesitará una frecuencia alta para representarse, al contrario que los cambios suaves, que están representados por las frecuencias bajas. El término espacial se refiere al hecho de que el nivel de cada píxel se calcula dependiendo de sus píxeles vecinos.

Los filtros se pueden dividir entre filtros lineales y no lineales. A su vez, los filtros lineales se pueden dividir en **filtros de paso baja** que eliminan las altas frecuencias, es decir, los cambios bruscos en el nivel de gris, **filtros paso alta**, que realizan la operación contraria, atenúan las bajas frecuencias y realzan las altas, y **filtros paso banda** que eliminan frecuencias intermedias. Como se puede deducir, si se eliminan los cambios bruscos de niveles de color (filtros paso baja) lo que se está haciendo es un difuminado de la imagen. Un filtro paso alta realza los cambios bruscos de gris, típicamente los bordes de los objetos, por lo que se suele utilizar como detector de bordes. En nuestro caso no vamos a utilizar los filtros paso banda.

La forma de operar de estos filtros se realiza mediante el uso de máscaras que recorren toda la imagen, en un proceso llamado **convolución**. En este proceso se recorre cada uno de los píxeles de la imagen superponiendo una matriz de dimensiones (2N+1)x(2M+1), cuyo centro coincidirá con el píxel que se está procesando. Para calcular los píxeles de la imagen resultante *g*, se utiliza la matriz de

convolución *h*, sobre la imagen original *f*. El valor de un pixel concreto de coordenadas (i,j) de la imagen resultante se obtiene de la siguiente forma:

$$g(i,j) = f \otimes h = \sum_{m=-M}^{M} \sum_{n=-N}^{N} f(i+m, j+n) * h(m+M, n+N)$$

Dependiendo de las matrices de convolución (también denominadas kernels) el resultado será uno u otro. Por ejemplo, para difuminar una imagen se puede utilizar una matriz cuyos elementos sean iguales y sumen 1. Este filtro se conoce como **promediado del entorno**. Mientras más grande sea la máscara (kernel) más alto será el efecto de difuminado puesto que estamos teniendo en cuenta píxeles más alejados del que estamos tratando.

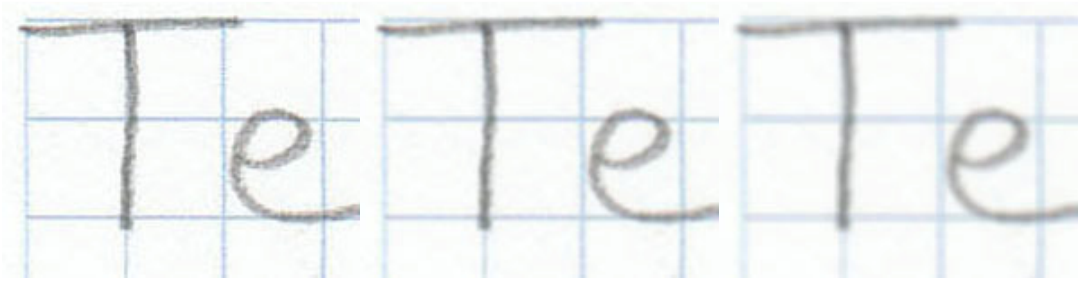

Figura 14: De izquierda a derecha: imagen original, imagen con filtro 3x3, imagen con filtro 5x5

La ventaja de este tipo de filtro es que es sencillo pero enturbia mucho los contornos. Para ello existe otro tipo de filtro con un mejor compromiso entre filtrado y enturbiamiento, como es el **filtro gaussiano**. Reemplaza cada píxel por una media ponderada de sus vecinos, es decir, no todos los tiene en cuenta por igual. En este caso, mientras más alejado estén del centro de la matriz de convolución, menos peso tienen en el filtro. Concretamente la matriz de convolución se obtiene con la siguiente fórmula:

$$g(x,y) = \frac{1}{2\sigma^2\pi} e^{-\frac{1}{2}\left(\frac{x^2+y^2}{\sigma^2}\right)}$$

Donde el punto (0,0) se considera el centro de la matriz. El tamaño de la máscara depende de σ, siendo la máscara de tamaño NxN, donde:

$$N = 2w + 1$$

$$w \geq 6\sqrt{2}\sigma \mid w \in \mathbb{N}$$

El filtro gaussiano conserva mejor los bordes de la imagen pero a costa de tener una máscara mayor y por tanto un mayor coste computacional. También se puede aplicar mediante convolución, al igual que el anterior.

Otro tipo de filtros son los paso alta, como se han comentado anteriormente. Nosotros los vamos a utilizar para la detección de bordes. Uno de los más utilizados es el operador de Sobel. Los kernel de este operador son los siguientes:

$$H_v = \frac{1}{4}\begin{bmatrix} 1 & 0 & -1 \\ 2 & 0 & -2 \\ 1 & 0 & -1 \end{bmatrix}$$

$$H_h = \frac{1}{4}\begin{bmatrix} -1 & -2 & -1 \\ 0 & 0 & 0 \\ 1 & 2 & 1 \end{bmatrix}$$

La primera matriz detecta los bordes verticales y la segunda los horizontales. Si sumamos el valor absoluto de las imágenes obtenidas al convolucionar ambos kernels, obtendremos la imagen de bordes. El tamaño del operador también afecta al resultado: Si el operador es pequeño localizará el borde con una precisión mayor pero esto hace que sea más sensible al ruido, por lo que podrá detectar bordes que no existen. Si se utiliza un operador mayor mejora la detección del borde ya que no le afecta tanto el ruido pero empeora la localización y aumenta el coste computacional.

El funcionamiento de estas matrices es intuitivo. Si se encuentran en una zona homogénea, al sumar todos sus elementos cero, el valor obtenido para un píxel será bajo. En cambio, si nos encontramos con un borde vertical una de las dos columnas de la matriz H_v tomará valores sensiblemente diferentes de la otra con lo que el valor absoluto será más alto que en el caso homogéneo. Por ejemplo, en el caso extremo que pasemos de blanco a negro, la columna primera valdrá (1*255+2*255+1*255)/4 = 255 y la última 0, por lo que en la zona del borde obtendremos un punto blanco. En caso de que pasemos de negro a blanco, el valor total será -255, pero como estamos tomando los valores absolutos el resultado es el mismo.

El problema de este filtro es que si el borde no es totalmente vertical u horizontal, los valores resultantes del borde serán inferiores. Esto es debido a que el cambio de intensidad máximo se consigue en la dirección del vector gradiente, es decir, perpendicular al borde. Si nos damos cuenta este filtro para detectar el borde se basa en la primera derivada, es decir, su valor depende de la intensidad del cambio de color. Mientras mayor sea el cambio mayor será el valor resultante. Por tanto habría que considerar un umbral a partir del cual consideramos que es borde o no. Esto es un inconveniente ya que este umbral dependerá de la zona de la imagen en la que nos encontremos, etc. Para solucionar este problema vamos a utilizar un filtro basado en la segunda derivada, ya que existirá un borde en aquel punto en el que la segunda derivada sea 0 (máximo de la primera derivada).

Este nuevo filtro es conocido como **Laplaciana**. Su kernel es el siguiente:

$$H = \begin{bmatrix} 0 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 0 \end{bmatrix}$$

La ventaja de este operador es que proporciona casi siempre bordes cerrados y de un píxel de espesor. El inconveniente es que es muy sensible al ruido con lo que la detección de los bordes se ve comprometida. Para evitar el problema del ruido lo que se hace es combinarlo con el operador de filtrado gaussiano. Primero se elimina el ruido que pueda presentar problemas y a continuación se realiza la detección de los bordes. Como todos los procedimientos tiene ciertos inconvenientes, como son el alto coste computacional debido al operador gaussiano y un pobre comportamiento en las esquinas.

Nosotros utilizaremos el operador Laplaciana junto con un filtro gaussiano para obtener los bordes del ejercicio escaneado. De esta forma podemos detectar el código de barras, la inclinación con la que se ha escaneado (para su posterior corrección), etc.

2.8.4. Procedimiento "Varita mágica" o "Magic wand"

La "Varita mágica" es una herramienta para seleccionar las partes de una imagen que se parecen entre sí y están adyacentes. El concepto "parecerse entre sí" viene determinado por la diferencia en el tono de los píxeles y es un parámetro de entrada al procedimiento. Esta herramienta se puede utilizar para segmentar partes de una imagen. En nuestro caso la vamos a utilizar para obtener el borde del código de barras, como se comentará en el apartado 3.4.2.

Según (González Jiménez, 1999), el objetivo de este procedimiento es segmentar la imagen tal que:

- Los píxeles de una región segmentada deben ser similares entre sí.
- Los píxeles de distintas regiones deben tener atributos diferentes.

El algoritmo utilizado es el siguiente:

1. Se toma como punto de partida un píxel y se inserta en el conjunto de píxeles a procesar
2. Mientras queden puntos en el conjunto de píxeles a procesar:
    a. Se coge un punto de dicho conjunto
    b. Si el punto cumple el requerimiento, no ha sido procesado y no está dentro del conjunto de píxeles a procesar, se inserta en el conjunto a devolver. Además, se insertan en el conjunto de píxeles a procesar sus vecinos siempre que no se encuentren ya o no hayan sido procesados.
    c. Se borra el punto escogido del conjunto de píxeles a procesar.

En nuestro caso el requerimiento será que el píxel sea suficientemente parecido en cuanto al nivel de gris a la media de los puntos que en ese momento pertenezcan al conjunto. Por cada píxel que se añada, hay que volver a calcular la media.

Un ejemplo del uso de la varita se puede ver en la Figura 15, donde como píxel inicial se ha tomado uno interior de la parte blanca. La región segmentada muestra una línea discontinua en su borde.

Figura 15: Ejemplo de la selección con la "Varita mágica"

2.9. Expresiones regulares

Las expresiones regulares son una forma de describir cadenas de caracteres. Cada una de ellas es un patrón formado por caracteres y símbolos y se construyen de igual forma que las operaciones aritméticas, es decir, mediante operadores que combinan más expresiones.

En el sistema se van a utilizar las expresiones regulares para generar palabras de forma automática. Estas expresiones están definidas en la API de Java (Sun Microsystems, Inc., 2008). Otro uso que tienen este tipo de herramientas es para la comprobación de que una cadena cumpla un patrón determinado.

Existe un conjunto de símbolos a los que se les da un significado para realizar las diferentes operaciones:

```
\ ^ $ . [ ] { } | ( ) * + ?
```

Los significados de cada uno de ellos se explican a continuación:

x	El carácter `x`
[abc]	`a`, `b`, o `c` (alternativa).
[^abc]	Cualquier carácter excepto `a`, `b`, o `c` (negación)
[a-zA-Z]	Cualquier carácter desde la `a` a la `z` o desde la `A` a la `Z` (rango)
[a-z&&[def]]	`d`, `e`, o `f` (intersección)
[a-z&&[^bc]]	Cualquier carácter desde la `a` a la `z` excepto `b` y `c`: `[ad-z]` (substracción)
[a-z&&[^m-p]]	Cualquier carácter desde la `a` a la `z` excepto los que se encuentren entre la `m` y la `p`: `[a-lq-z]`(substracción)
.	Cualquier carácter
***X*?**	*X*, una o ninguna vez
X*	*X*, cero o más veces
***X*+**	*X*, una o más veces
***X*{*n*}**	*X*, exactamente *n* veces
***X*{*n*,}**	*X*, por lo menos *n* veces
***X*{*n*,*m*}**	*X*, por lo menos *n* veces pero no más de *m*
XY	*X* seguido de *Y*
X*|*Y	*X* ó *Y*
(X)	*X* capturado como grupo

El carácter '\' se utiliza para especificar caracteres especiales (\t para la tabulación, etc.) o bien para tomar como literal los operadores (por ejemplo '*' es el carácter asterisco y no el operador 'cero o más veces'). No vamos a profundizar en el funcionamiento de estos caracteres especiales puesto que no vamos a utilizarlos.

3. Diseño del sistema de apoyo a la escritura

En el capítulo actual se comentarán los algoritmos implementados en el sistema, el enfoque que se le ha dado a los problemas, el modelado de diferentes elementos del programa, etc., desde una forma a alto nivel, es decir, independiente del lenguaje de programación utilizado en su implementación, sin ningún tipo de código. Además se comentará la interfaz gráfica de modo que cualquier usuario sepa manejar completamente el programa.

3.1. Entrada de datos

El sistema que aquí se presenta utiliza dos tipos de entrada de datos: en formato digital a través de una pantalla escribible (por ejemplo un TabletPC) o escaneados desde papel. Estos modos de realizar los ejercicios hace que los datos se nos presenten de diferente forma.

Si utilizamos un TabletPC, se obtienen una serie de puntos provenientes de los eventos del ratón/lápiz que lanza el sistema operativo. Por cada evento obtenemos tres datos:

- El tipo del evento, que puede ser “botón pulsado”, “botón soltado” o bien “arrastre”. El evento de arrastre indica que está pulsado el botón del ratón mientras se ha movido. Los equivalentes con el lápiz del TabletPC se producirían al tocar el lápiz la pantalla táctil, al levantarlo de la misma o bien al escribir con él sobre la pantalla.
- El punto de la pantalla en el que ha ocurrido el evento. En realidad el punto es relativo al componente que ha producido el evento, pero los ejercicios se realizan a pantalla completa, por lo que coinciden exactamente con el ejercicio.
- El momento en el que se ha producido el evento. Aquí no nos interesa saber la fecha del evento, sino el momento relativo al comienzo de la realización del ejercicio. Eso se consigue guardando la fecha en la que se empezó y restándosela a la que se nos proporciona en el evento.

Estos datos se almacenan en un archivo de log (además de otros), que se utilizará para corregir el ejercicio. La forma en que se guarda el log es la siguiente: cada evento genera una línea. Cada línea tiene 4 parámetros. El primero es el tipo de evento, que en el caso que nos interesa será un 1 para “botón pulsado” o su equivalente con el lápiz, un 2 para el evento de arrastre y un 3 para “botón soltado”. El segundo y tercer parámetro son las coordenadas X,Y donde se ha producido el evento. Y por último se guarda los milisegundos que han pasado desde el inicio del ejercicio. Un ejemplo de un segmento del log se puede ver en la Figura 16:

```
1 115 293 30827
2 117 291 31160
3 117 291 31160
1 125 280 31236
2 126 275 31569
2 127 269 31667
2 127 265 31762
3 127 261 31921
```

Figura 16: Ejemplo de log

Una vez obtenido el log, se puede realizar la corrección del ejercicio, lo cual veremos en un apartado posterior.

En el caso de los datos obtenidos mediante el escaneo del ejercicio en papel, el proceso es totalmente diferente. Para obtener lo escrito por el alumno hay que recurrir al tratamiento de imágenes. A continuación se enumeran los pasos que se siguen:

Una vez que se tiene la imagen escaneada hay que filtrar la imagen para que la segmentación de lo escrito por el alumno se pueda realizar con el menor error posible. La zona de escritura está definida por unas marcas para facilitar la segmentación, ya que de otra forma hay demasiados colores y se incurriría en errores de detección. Así sólo analizamos la zona donde debe haber escritura que contendrá la cuadrícula (que debe ser de un mismo color, o por lo menos de un color muy parecido) y lo escrito por el alumno.

- Por cada zona donde el alumno tenga que escribir se obtiene una imagen, basándonos en las marcas que se han impreso. A esta imagen se le realiza un filtrado gaussiano (explicado en la sección 2.8.3) para que la diferencia entre los píxeles del mismo color adyacentes sea mínima. Esto ocurre en mayor medida si el ejercicio se realiza con lápiz, donde aunque el color teóricamente debe ser el mismo, dependiendo de la presión, de la zona de la mina, etc., el color puede ser ligeramente distinto. Con este filtro conseguimos igualar en la medida de lo posible estos colores.
- A continuación se aplica un filtro de brillo y contraste (véase sección 2.8.2). El color de la cuadrícula es un azul claro, y el color de la escritura del alumno suele ser más oscuro, ya sea lápiz, bolígrafo, etc., por lo que al realzar la imagen la cuadrícula se fundirá con el fondo al aumentar el brillo y sólo

quedará la escritura del alumno en un color bastante oscuro al aumentar el contraste.

- Seguidamente se utiliza un umbral para binarizar la imagen, es decir, todo lo que esté por encima de un punto se considerará blanco y lo que esté por debajo, negro.
- A esta imagen binaria se le calcula el esqueleto y se obtiene la escritura del alumno con un píxel de grosor. Con esto ya estamos en disposición de compararlo con la solución perfecta y realizar la evaluación del mismo.

Nótese que mediante el tabletPC se puede obtener un dato más que mediante el escaneo: el tiempo. Este dato puede ser muy útil a la hora de evaluar otros aspectos además de la propia escritura, como la velocidad, etc., que también son importantes. El tiempo también da una medida de la capacidad motora del alumno ya que, como es lógico, es mucho más difícil realizar bien un ejercicio en poco tiempo. De esta forma el profesor tiene más información sobre los alumnos y por tanto se podrán tomar más medidas para la mejora de las habilidades de los estudiantes.

En la Tabla 3 se muestra un resumen de las ventajas e inconvenientes de cada tipo de entrada de datos:

	Ventajas	Inconvenientes
Papel y escáner	- Mínimo impacto para el alumno - Facilidad de uso	- Altos recursos de CPU - Se pueden producir errores en el proceso de detección de la escritura
TabletPC	- Corrección de los ejercicios de forma inmediata - Se puede obtener el tiempo	- Requiere un proceso de adaptación - Hay que realizar más presión sobre la pantalla que sobre un papel

Tabla 3: Ventajas e inconvenientes de los dispositivos de entrada

3.2. El modelo de usuario

El modelo de usuario almacena el estado y nivel de conocimiento del alumno y tiene dos funciones básicas en el sistema:

- Informar al educador del nivel del alumno, es decir, qué parte necesita mejorar y qué parte domina
- Ser la información base para la generación automática de los ejercicios adaptativos

Para guardar toda esta información se utilizan los campos mostrados en la Tabla 4. También se comenta para qué se utiliza cada uno de ellos.

Campo	Descripción
Nombre	En este campo se guarda el nombre
Apellidos	Almacena los apellidos
Fecha de nacimiento	Es la fecha de nacimiento
Clase	La clase en la que se encuentra matriculado
Nivel	El nivel que tiene
Paso de workflow	El paso de workflow en el que se encuentra
Anchura	Valoración de la anchura de los ejercicios, del 0 al 100
Altura	Valoración de la altura de los ejercicios, del 0 al 100
Tamaño de letra Inconsistente	Valoración de la diferencia del tamaño de las letras. Idealmente no debería haber diferencia, y se asignaría 100 si lo hace bien ó 0 en caso contrario.
Legibilidad	Valoración de la legibilidad del texto escrito, del 0 al 100.
Aciertos	Porcentaje de aciertos de lo que se ha escrito (es decir, que lo que se ha escrito esté en posición correcta).
Acierto por letra [aA..zZ]	Por cada una de las letras del abecedario, existe un campo como este. Especifica el porcentaje de acierto en la letra correspondiente.

Tabla 4: Campos del modelo de usuario

El modelo de usuario se actualiza de dos formas dependiendo del campo. Los campos de información personal del alumno, como pueden ser el nombre, los apellidos, la fecha de nacimiento, etc., se modifican manualmente (estos campos se muestran en rojo en la tabla), mediante un formulario específico. Hay otros campos, como el nivel o el paso de workflow que se pueden modificar manualmente (esto se suele hacer al principio, cuando se ha insertado el alumno) o bien mediante el procesado de la información que se ha obtenido al corregir los ejercicios (se muestran en azul en la tabla). Por último, existen campos que sólo se modifican conforme el niño va progresando en su aprendizaje, según haga mejor o peor los ejercicios (se muestran en verde).

A continuación se va a explicar cómo se modifican los campos a partir de un ejercicio. Por tanto, sólo se modificarán aquellos que estén en azul o en verde en la Tabla 4.

En el modelo de usuario sólo se guarda la información de los últimos *n* ejercicios, donde *n* es un número especificado por el educador. Esto representa el conocimiento actual del alumno. También se almacenan los resultados de todas las correcciones realizadas, para poder ver el progreso del niño.

- **Nivel**: Se actualiza al realizar cualquier ejercicio. Si a un alumno se le asigna un modelo de un determinado nivel, ya sea manual o automáticamente, se supone que es porque ya tiene los conocimientos necesarios. Por tanto, si a la hora de corregir un ejercicio, el nivel de éste es mayor al almacenado en el modelo de usuario, se actualiza al nivel del ejercicio realizado.

- **Paso de Workflow**: Este dato se puede actualizar manualmente, como se ha comentado antes, ya sea para asignar el alumno al primer paso de workflow (esto es inevitable, ya que el sistema debe saber por dónde empezar), o para cambiarlo "artificialmente". Además, **se cambia automáticamente al siguiente paso después de corregir cada ejercicio**, siempre que se cumplan las condiciones establecidas, es decir, que haya realizado los mínimos ejercicios requeridos en el paso actual, que la valoración del modelo de usuario supere o iguale el umbral especificado, etc.
- **Anchura, altura, tamaño inconsistente de letra, legibilidad, aciertos y aciertos por letra**: Todos estos datos se actualizan de la misma forma. Cada vez que corrige un ejercicio, se obtienen los resultados por cada uno de estos campos (véase apartado 3.4). A continuación se hace una media con los datos ya guardados (esta media dependerá de un parámetro especificado por el educador, que indica el número de ejercicios para realizarla) y se almacena.

Este modelo se ha implementado en una Base de Datos con la siguiente arquitectura:

Por una parte tenemos una tabla de alumnos, donde se guarda información relativa al mismo, como puede ser el nombre, los apellidos, la fecha de nacimiento, etc. También se almacena el qué nivel se encuentra el alumno y en qué paso del workflow en caso de estar en alguno. Además existen dos tablas donde se van guardando los resultados: '*students_history*' y '*students_summary*'. En *student_history* se guardan todos los ejercicios resueltos por el alumno y su evaluación. Por cada letra se guarda un valor que indica la capacidad del alumno para realizarla, además se guardan datos sobre la anchura de las palabras, altura, etc. En *students_summary* se guarda un resumen de los resultados del alumno en cada nivel, que son la media de los últimos *n* ejercicios de un determinado nivel, como hemos comentado anteriormente. A partir de la información proporcionada por estas tablas podemos saber la trayectoria de un alumno, si debe pasar de nivel, etc.

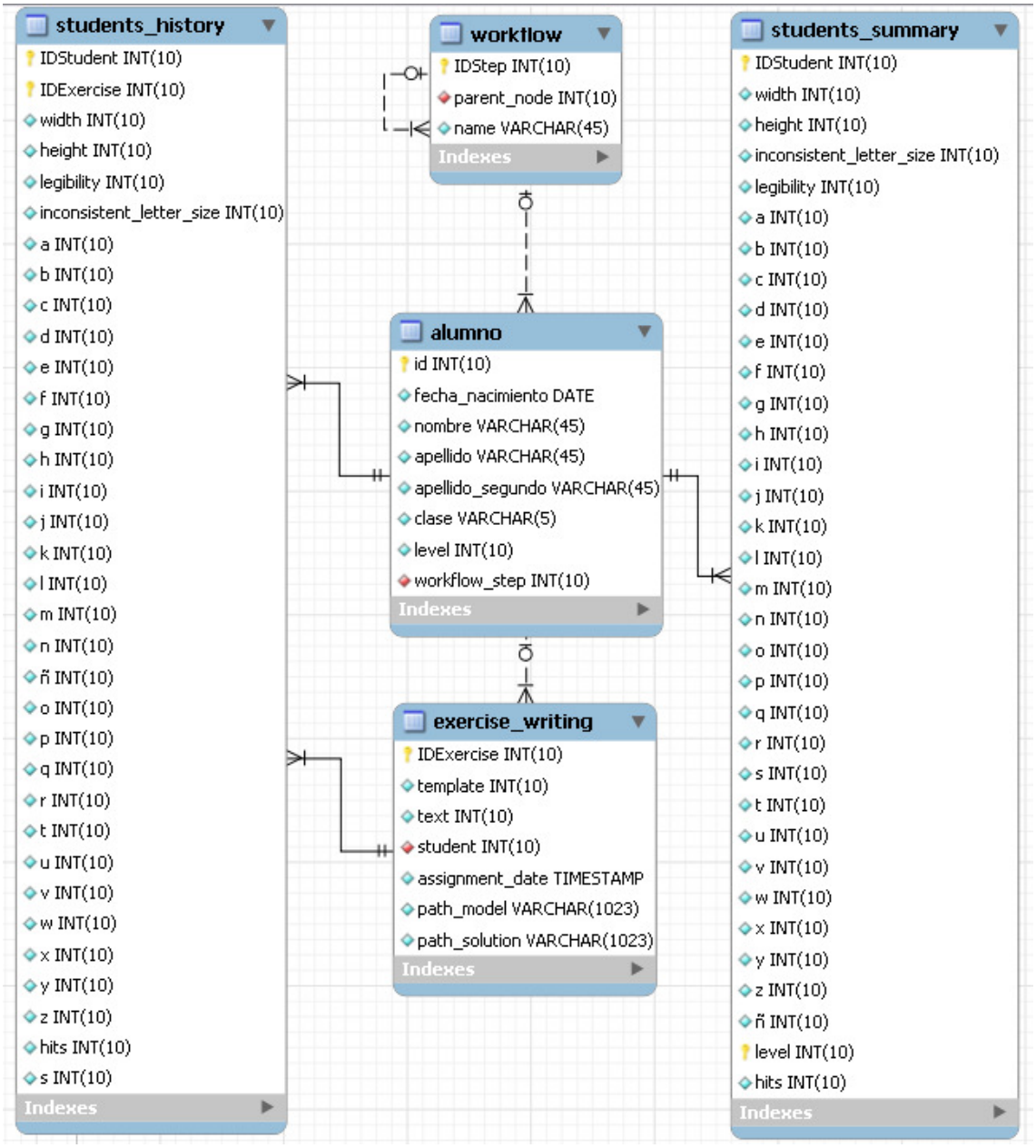

Figura 17: Esquema de la BD para el modelo de usuario

3.3. El modelo de tareas basado en plantillas y formatos

Este proyecto está diseñado para adaptarse a las necesidades de los alumnos de niveles muy diferentes, desde los que están aprendiendo a leer y a escribir hasta los que ya tienen la habilidad desarrollada pero requieren de una mejora de su caligrafía. Esto requiere que la generación de ejercicios sea lo suficientemente flexible y que todos los estudiantes tengan ejercicios adecuados a su nivel.

Hay que tener en cuenta que cada alumno va a realizar bastantes ejercicios de un mismo nivel, hasta que los realice correctamente. En ese caso, la dificultad no va a cambiar y lo único que será diferente será el contenido, es decir, las letras/palabras del

ejercicio a realizar. Para este menester se ha diseñado un sistema de plantillas, donde se define la dificultad del ejercicio y su formato. Cada plantilla lleva asociado los siguientes datos:

- El tipo de fuente utilizada
- El tamaño de la fuente.
- El nivel del ejercicio (la dificultad para la que se ha diseñado).
- El formato, que define cuántas líneas de patrón va a haber, cuántas veces se ha de copiar cada patrón, el número de dibujos, etc.
- La configuración del papel (si es A4, A5, apaisado, los márgenes izquierdo y derecho, etc)
- El nombre de la plantilla
- El espacio entre líneas del ejercicio.

De esta forma sólo hará falta definir el texto de cada línea de patrón y elegir los dibujos a realizar. Si no se cambian los dibujos de un ejercicio a otro es posible obtener una respuesta negativa por parte del alumno a la hora de hacer el ejercicio si ya ha realizado uno con el mismo dibujo. La representación de una plantilla puede ser como sigue:

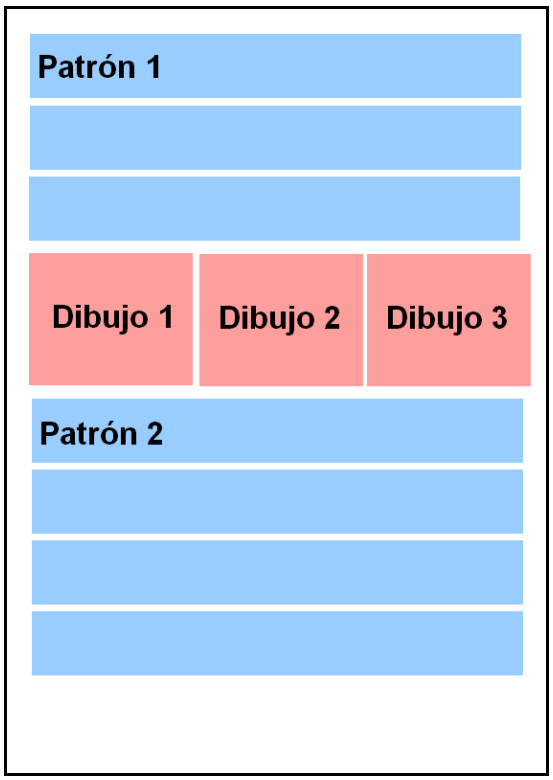

Figura 18: Ejemplo de plantilla

En la Figura 18 se muestra una plantilla con dos patrones y tres dibujos. El alumno deberá copiar los patrones en las líneas que le siguen. El primer patrón lo deberá copiar dos veces y el segundo tres veces.

Una vez definidas las plantillas necesarias se pueden generar los ejercicios, que simplemente consistirá en elegir una plantilla y definir el contenido.

3.4. El modelo de corrección

Al asignar un ejercicio a un alumno, se genera la solución paralelamente a éste. Este patrón nos va a servir para comparar la escritura del alumno con la solución perfecta. Dependiendo de la diferencia entre las dos, se valorará el trabajo del alumno. En la Figura 19 se puede observar un ejercicio y su solución.

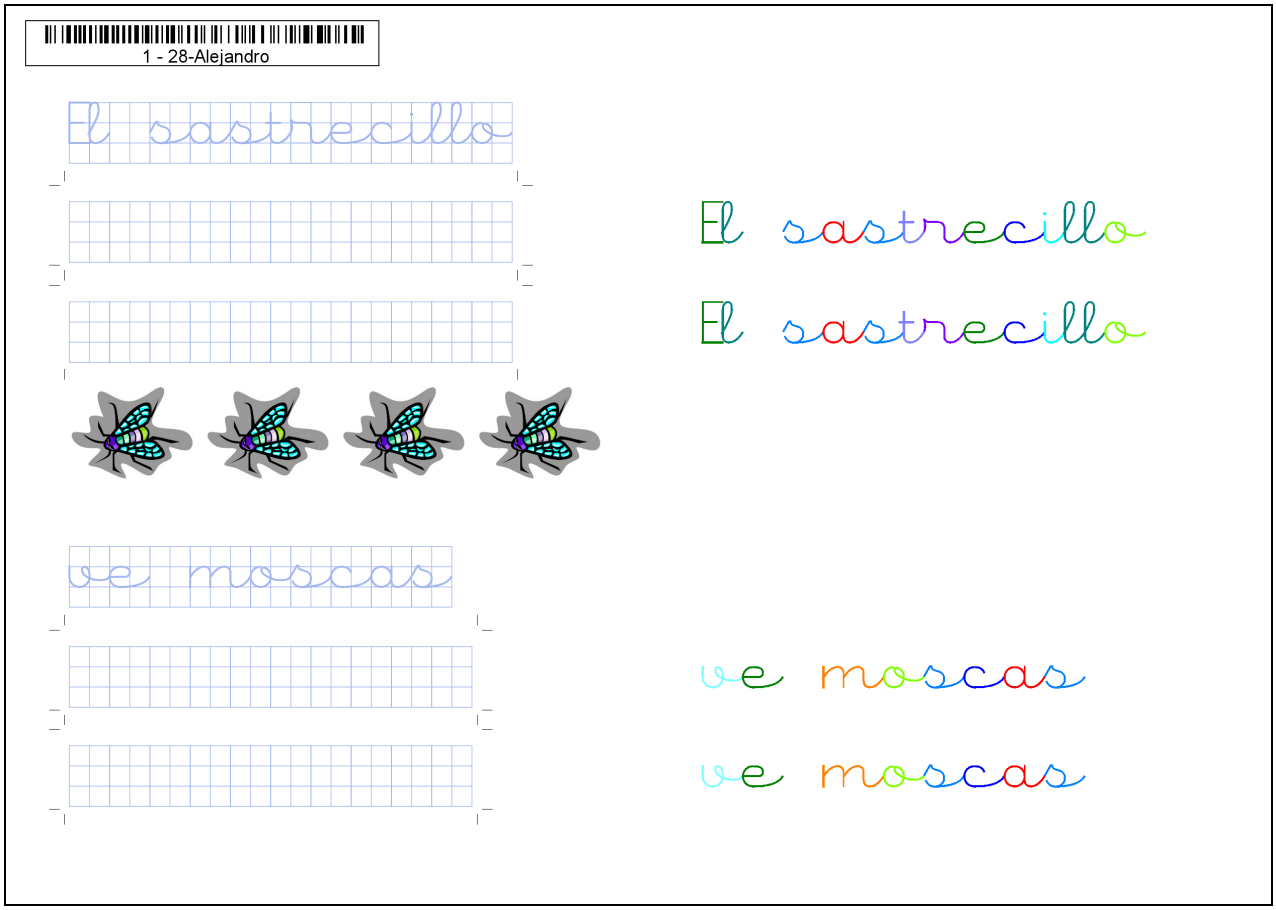

Figura 19: Ejemplo de ejercicio y solución asociada

En la solución se utilizan diferentes colores para las letras para conocer exactamente dónde empieza y termina cada una de ellas. De esta forma se puede evaluar cada una independientemente sin que intervenga el tipo de fuente utilizado. Esto es conveniente ya que puede haber alumnos que tengan problemas al escribir determinadas letras en concreto, por lo que habrá que tomar medidas. En general a

este alumno se le asignarán ejercicios que contengan letras con las que tiene dificultad.

Dependiendo del tipo de entrada de datos, la corrección es diferente.

Al usar el tabletPC para realizar el ejercicio se ha guardado el log donde encontramos los puntos por los que ha pasado el alumno. En un primer prototipo la corrección tiene dos partes. Por un lado se comprueban cuántos de los puntos que ha escrito el alumno son correctos. Para considerar un punto correcto debe estar encima de alguna de las letras, aunque se permite cierto error, que está parametrizado. A continuación se puede observar un ejemplo de un ejercicio realizado por un niño de 4 años. Los puntos verdes se toman como correctos y los rojos como erróneos.

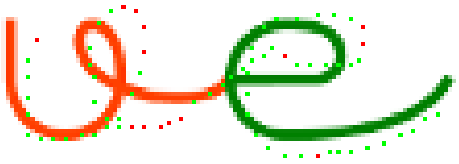

Figura 20: Ejemplo de corrección

Evidentemente esto nos da una información bastante pobre, ya que si el alumno ha realizado el ejercicio perfecto, el resultado será que el 100% de los puntos son correctos. Pero la implicación contraria no es cierta, es decir, si el 100% de los puntos que se han escrito son correctos, el ejercicio no tiene porqué estar perfecto. Por ejemplo, si sólo se completa la mitad de cada letra, etc. Por tanto añadimos una nueva información: por cada punto de cada letra de la solución se comprueba si existen puntos escritos que coincidan. Al igual que antes, se introduce cierto margen de error, con lo que lo que se comprueba es si existe un punto escrito en un radio X. La definición de distancia que se ha tomado para este caso es la distancia de Tchebichev ($D(p_1, p_2) = \max(|x_1-x_2|, |y_1-y_2|)$. A partir de esta información ya podemos saber qué parte del ejercicio se ha completado, qué letras hace mejor, cuáles peor, etc.

Para un ejercicio escaneado la corrección es la misma excepto que los puntos escritos no se obtienen del log, sino del esqueleto de la imagen segmentada. Se recorren todos los puntos y los que sean negros se consideran que es la escritura del alumno y los que sean blancos se considera parte del fondo.

En un segundo prototipo se calcula otro tipo de información. A partir de la solución, se calcula la media y la varianza de la coordenada X e Y de cada palabra. Para conseguir esto se realizan los siguientes pasos:

- A partir de las marcas de cada línea del ejercicio, se calculan las coordenadas de la misma.
- Por cada línea, se calcula dónde comienza cada palabra y dónde termina. De esta forma podemos calcular el píxel que se encuentra en el medio de dos palabras.
- Para cada palabra se calcula la media de las coordenadas de los píxeles de las letras.
- Se realiza otra pasada para calcular la varianza de los píxeles de cada palabra.

Esto se realiza tanto en la solución como en el ejercicio resuelto por el alumno. Comparando la media y las varianzas se pueden obtener diferentes conclusiones. Por ejemplo, si la varianza de la coordenada Y de una palabra es menor que la de la solución perfecta, se puede suponer que las letras tienen menos altura de la que deberían. Si ocurre lo mismo con la coordenada X, significa que la palabra es más estrecha. Para el caso contrario ocurre lo mismo: si la varianza es mayor, significa que la palabra en ese eje es más amplia que la solución. Dependiendo de las diferencias entre las medias y las varianzas se obtiene una puntuación del 0 al 100 tanto para la altura como para la anchura.

Otros datos que se calculan son la legibilidad y el tamaño inconsistente de letra. La legibilidad se calcula a partir de los aciertos, de los aciertos por letra y de la puntuación del ancho y el alto. Como futuros trabajos se piensa mejorar el algoritmo para que el valor calculado sea independiente de la escala y posición de lo escrito por el alumno. El tamaño inconsistente de letra se calcula de la siguiente forma:

- Se compara la desviación típica en altura de cada palabra escrita por el alumno con la varianza de la solución perfecta. Se obtiene un porcentaje, que será la escala de lo realizado por el alumno respecto a la solución.
- Se realiza una media de los porcentajes obtenidos.
- Se calcula la desviación típica de los porcentajes, lo cual nos indica cuánto cambian de altura las letras que se han escrito. Si las escribe siempre con el mismo tamaño, aunque sea muy diferente del original, la desviación típica se acercará a cero, lo que indicaría una muy buena puntuación en este apartado.

Para obtener el valor que se guarda en el modelo de usuario a partir de la desviación típica calculada (*dtc*), se utiliza la siguiente fórmula:

$$valor = \begin{cases} 100 - \frac{100}{x} * dtc, & \text{si } dtc \in [0, x] \\ 0, & si\ dtc \in (x, \infty) \end{cases}$$

Donde *x* es un parámetro que indica el umbral del porcentaje a partir del cual el valor calculado va a ser cero.

3.4.1. Segmentación del ejercicio escaneado

Cuando un ejercicio se realiza en papel, la introducción al sistema se hace mediante escáner. Esto quiere decir que lo que recibe el sistema es una imagen (que sabemos cómo está compuesta) y hay que detectar la escritura del alumno, el código de barras, etc.

Lo primero que se hace es calcular el ángulo con el que se ha escaneado. Se necesita una imagen totalmente horizontal para poder compararla con la solución generada al asignar el ejercicio al alumno. Dado que es muy complicado escanear un papel sin la mínima rotación, se realiza esta operación para un funcionamiento óptimo del sistema de evaluación automático. Para llevar a cabo esta operación lo primero que se hace es calcular la imagen de bordes. Para ello se utiliza un filtro Gaussiano para suavizar la imagen y a continuación un filtro Laplaciana de paso alta. Como se comentó en el apartado 2.8.3, el filtro gaussiano es necesario puesto que la Laplaciana es muy sensible al ruido por lo que sin él la detección de bordes se vería comprometida.

A continuación se calcula la inclinación del borde. Esto se puede hacer mediante la línea que se crea entre el papel escaneado y la tapa del escáner o bien mediante el borde del código de barras en caso de que la tapa sea blanca y no haya diferencia entre ésta y el papel. El procedimiento que se sigue es muy simple: una vez detectado el borde, éste se va siguiendo hasta el final. Tenemos las coordenadas iniciales (x_1, y_1) y las finales (x_2, y_2). Con esos datos y la función arcotangente se calcula el ángulo de inclinación mediante la fórmula

$$\theta = \tan^{-1}\left(\frac{y_2 - y_1}{x_1 - x_2}\right)$$

Al numerador se le ha cambiado el orden de los sumandos para que el signo obtenido sea el adecuado para la función de Java que realiza la rotación de la imagen.

Una vez está el ejercicio completamente recto, decodificamos el código de barras, lo cual se explica en el siguiente apartado. De ahí se obtiene el identificador del ejercicio. Con ese dato se puede leer el ejercicio desde el disco duro y obtener las marcas que delimitan la zona donde debe escribir el alumno. Por comparación de los códigos de barras (del escaneado y el del ejercicio original) se obtiene la relación de tamaño entre los dos. Conociendo dicha relación y los delimitadores en el ejercicio original, se puede obtener una aproximación muy buena para buscar los delimitadores en el ejercicio escaneado. De esta forma se minimiza la zona de búsqueda y se evita

detectar otro tipo de trazos en caso de que el alumno se haya salido de la zona delimitada.

Una vez hemos detectado los delimitadores del ejercicio escaneado, obtenemos las imágenes de cada una de las líneas. Cada imagen se reescala al tamaño de la imagen original para poder realizar la comparación. A continuación se cambia el contraste y brillo para obtener únicamente lo escrito por el alumno. Sólo queda umbralizar la imagen (lo que esté por encima de un determinado valor se considera blanco y lo que esté por debajo, negro) y calcularle el esqueleto, que será una línea que pasa exactamente por el centro del trazo que ha realizado el alumno. Con el esqueleto tenemos todos los datos para realizar la comparación.

3.4.2. Detección y decodificación del código de barras

En este apartado explicaremos cómo se realiza el tratamiento de la imagen para la detección y decodificación del código de barras.

En primer lugar, se puede suponer que el código de barras se encuentra en la parte superior izquierda de la imagen, ya que así es como se generan los ejercicios. En segundo lugar, se va a suponer que no existe ningún trazo ni escrito entre el borde del papel y el código de barras.

Con la imagen de bordes ya calculada, se empieza desde la parte superior del ejercicio a comprobar píxel a píxel hacia abajo si existe un borde. Para que siempre se encuentre el código de barras, se comienza en el 12% del ancho, es decir, si el ejercicio tiene 1000 píxeles de ancho, se comenzará con x = 120. Si hemos comprobado el 10% de los píxeles verticales y no se ha encontrado un borde, nos movemos hacia la derecha y realizamos la misma comprobación. En algún momento se deberá encontrar el borde del código de barras. En caso contrario, se devolverá null.

Una vez encontrado el código de barras, se obtienen todos los píxeles del borde mediante la función “varita Mágica” explicada en el apartado 2.8.4. Al tener todos los píxeles podemos obtener los vértices, ya que se calculan con las coordenadas mínimas y máximas de *x* e *y* respectivamente. El vértice superior izquierdo se creará con la mínima coordenada de *x* y la mínima de *y* que se encuentre entre los puntos del borde. El vértice superior derecho, será la mínima de *x* y la máxima de *y*. El vértice inferior izquierdo se calcula con la máxima de *x* y la mínima de *y*. Por último, como el lector podrá suponer, el vértice inferior derecho se calcula con la máxima de *x* y la máxima de *y*. Para poder comprender esto correctamente hay que tener en cuenta que en una imagen de NxM píxeles la coordenada (0,0) es la superior izquierda y la (X-1,M-1) es la inferior derecha.

Si recortamos la imagen por los vértices que acabamos de calcular, la resultante contendría los bordes del código de barras, cosa que no queremos. Para solucionar

este problema, desde la esquina superior izquierda se traza una línea imaginaria de 45º bajo el eje horizontal, tal y como se ve en la Figura 21. Siguiendo esta línea se comprueba cuándo acaba el borde y de ahí obtenemos el vértice interior del código de barras. Se realiza la misma operación desde el borde inferior derecho, pero esta vez en sentido contrario. Como la imagen ya fue rotada, con la coordenada superior izquierda y la inferior derecha podemos obtener el código de barras.

Figura 21: Sentido de búsqueda para eliminar el borde

Una vez tenemos la imagen completa sin borde, como sabemos la proporción del código, tomamos tres líneas horizontales, las cuales serán "cortes" del mismo código de barras. Una al 15% de la altura de la imagen obtenida, otra al 30% y otra al 45%. De esta forma obtenemos el mismo código de barras en tres zonas diferentes. Esto se hace porque puede tener ruido y tenemos que filtrarlo. Una vez obtenida las tres líneas se hace una media de ellas. Con la media ya podemos decodificarlo convenientemente.

El código de barras utilizado se explica en la sección 2.6. Si no se ha leído dicha sección, se recomienda encarecidamente hacerlo antes de seguir ya que de otra forma no se comprenderían procedimientos que aquí se explican.

Teniendo en cuenta el formato del código de barras, lo primero que se hace es calcularle los bordes por convolución a la media obtenida con anterioridad. En realizad la imagen que vamos a tratar tiene una altura de un píxel, pero la operación se realiza de la misma forma que si se tratase de otro tipo de imagen, pero con el kernel [1, -2, 1] en vez de uno cuadrado. Para saber más sobre la convolución véase el apartado 2.8.3.

Lo primero que se hace es calcular el ancho de las zonas. Hay que tener en cuenta que las zonas del código de barras (espacio y barra) van intercaladas empezando por un espacio. El primer espacio es un espacio en blanco que no se debe tener en cuenta. A continuación sabemos que los siguientes seis anchos corresponden al carácter de inicio y también sabemos que en cualquiera de los casos (existen tres caracteres de inicio dependiendo del código que se vaya a utilizar) la suma de sus anchos es 11. Por tanto dividimos el ancho en píxeles de los seis primeros anchos entre 11 y obtenemos el ancho mínimo, que va a ser el máximo común divisor de los demás (se recuerda que los símbolos pueden tener cuatro anchos diferentes, múltiplos del más pequeño). Una vez que hemos obtenido el ancho mínimo en píxeles, se dividen todos los demás anchos por este número, con lo que el resultado de cualquiera de

ellos debe ser un número entero entre 1 y 4, tal y como se especifica en la tabla del código de barras Code128.

El único problema que puede surgir a la hora de normalizar los anchos es que uno de ellos, debido a la absorción de tinta por el papel, problema en el escaneado, etc., se quede entre dos posibilidades. Por ejemplo, si el ancho mínimo son 4 píxeles y uno de ellos ocupa 5, se puede suponer que el ancho es 1, o si ocupa 7 píxeles el ancho sería 2. ¿Pero qué pasa si el ancho es 6? ¿Se asigna al 1 o al 2? Este caso no se trata ahora mismo, sino a la hora de comprobar el carácter. Los caracteres tienen una distancia mínima de 2 entre sí, es decir, si se componen de 6 dígitos, no hay ninguno en el que la suma de las diferencias de dígito a dígito sea menor que 2. Por tanto, si un carácter no se encuentra se escoge el más cercano. Hay que tener en cuenta que en cualquier caso el error del ancho no va a ser mayor que 1, es decir, si el ancho es 1, no se detectará más de 2, o si el ancho es 2 se podría detectar 1 ó 3.

Un ejemplo de este problema sería el siguiente. El carácter de la 'a' se codifica como 121124. Si por un problema en el escaneo se detecta 121123, que no existe, las alternativas serían 221123, 111123, 131123, 122123, 121223, 121113, 121133, 121122 y 121124. Debido a la distancia mínima comentada anteriormente sólo existe el último, el 121124, por lo que será el elegido.

Cabe destacar que este sistema sólo corrige un error por carácter, si bien es cierto que si se selecciona el carácter incorrecto, luego el CRC no coincidirá.

Una vez detectados todos los caracteres, basta comprobar el CRC de la forma en la que se explica en el apartado 2.6 para comprobar que todo es correcto.

3.5. Generación automática de ejercicios

A la hora de generar un ejercicio hay que escoger el que mejor se adapte a cada alumno. Para ello se tienen en cuenta los metadatos del paso de workflow en el que se encuentra el alumno, los metadatos de los modelos y los de las plantillas, además de los datos del modelo de usuario.

Los metadatos pueden ser de tres tipos: Simples, intervalo de enteros o conjunto de valores. Para saber más de cada uno de ellos consúltese la sección 3.7.5.

Existen dos formas de generación de ejercicios: manual y automáticamente.

En la forma manual se evalúa cada alternativa y se le da a elegir al usuario qué modelos o qué plantillas utilizar junto con un valor que informa de lo adecuado que es para el alumno. La razón de esta forma estriba en que puede haber motivos por los que el educador prefiera otros modelos y que no se pueden averiguar mediante el modelo de usuario.

En la forma automática simplemente se especifica el número de ejercicios a generar para cada alumno y el sistema los asignará sin preguntar nada al usuario.

Lo primero es comprobar en qué paso se encuentra el alumno y obtener los metadatos correspondientes. Una vez realizada esta consulta, se siguen los pasos que a continuación se describen:

1.- Se comprueban si existen modelos que tengan los metadatos requeridos por el paso correspondiente.

- Si existen, se evalúa cada uno de ellos y se les asocia un valor que nos indica cuánto es de adecuado el modelo para el alumno (ver sección 3.5.1). Si estamos en modo automático, se asocian al alumno. Si no, se muestran para que el profesor pueda seleccionar los que crea conveniente y posteriormente se asocian los elegidos. Finaliza el proceso.
- Si no existen modelos, se buscan plantillas que cumplan los requisitos del paso de workflow mediante los metadatos, es decir, se sigue en el paso 2.

2.- Se comprueban si existen plantillas que se puedan asignar en el paso de workflow actual.

- Si existen y estamos en modo automático, se seleccionan tantas plantillas como ejercicios se hayan especificado, a menos que no haya suficientes plantillas. En este caso se seleccionarán todas las disponibles. Si no estamos en modo automático, se muestran las disponibles al usuario para que escoja las que desee según su criterio.
- A continuación, a las elegidas se les asigna el texto automáticamente. Para ello se generan todas las palabras posibles en el paso de workflow, y se evalúa cada una de ellas, es decir, se les asigna un valor numérico que indica la idoneidad según su modelo de usuario. Para ello se tiene en cuenta el tamaño de la palabra, ya que el objetivo es maximizar el valor educativo del texto. Por tanto, se escogen las palabras que tienen mayor valor medio por letra. Cabe destacar que por cada palabra se tiene en cuenta el espacio que le sucede, ya que de otra forma podría ocurrir que en el texto hubiese muchas palabras cortas y por tanto, muchos espacios ocupando sitio, lo cual puede que no fuese lo óptimo. Este proceso se explica con detalle en el siguiente subapartado.

3.5.1. Valoración de los modelos/palabras

Para poder asignar automáticamente un modelo a un alumno o escoger el texto que mejor se adapte a sus necesidades hace falta una función que nos dé un valor de cada uno de ellos.

Lo primero que se debe hacer es consultar en la base de datos, en el modelo de usuario, los valores de las distintas letras del alumno. Estos valores van del 0 al 100, donde el 0 indica que el alumno no realiza bien la letra y 100 que la escribe perfectamente.

Para valorar una palabra se comprueban cada una de las letras de la misma. Por cada letra se resta a 100 el valor que tiene el alumno en el modelo de usuario para dicho carácter (que a su vez puede tomar los valores del intervalo [0-100]). Una vez sumados todos los valores, se dividen por el número de letras más uno de dicha palabra. Esto es fundamental para optimizar el ejercicio ya que cada carácter de espacio no tiene valor educativo, pero cada palabra debe llevar uno detrás, por lo que es más recomendable una palabra de 5 letras cuyos caracteres tengan de media un valor de 50, que una palabra de dos letras que tengan 52 de valor medio. Por ejemplo, supongamos que el alumno tiene los siguientes valores en el modelo de usuario: a=35, i=40, m=43; y que las palabras que se pueden escoger son 'mima' y 'mi'. Si sólo tenemos en cuenta las letras de cada palabra, el valor de 'mi' sería: (43+40)/2= 41.5. Y el valor de 'mima' sería: (43+40+43+35)/4 = 40.25. Pudiera parecer que la palabra 'mi' tiene más valor por letra, pero si tenemos en cuenta el espacio que hay detrás de cada palabra, los resultados son para 'mi': (43+40+0)/3= 27.67, y para 'mima': (43+40+43+35+0)/5=32.2. Claramente, la palabra 'mima', aún teniendo menos valor por letra, tiene más valor por "espacio ocupado en el ejercicio".

Cuando se ha obtenido el valor medio por carácter de cada palabra, se ordenan y se van escogiendo de mayor a menor hasta completar el ejercicio. En caso de que una palabra no quepa, se intenta seleccionar la inmediatamente inferior. Así hasta que se encuentre la palabra correspondiente.

En el caso de los ejercicios/modelos, que ya tienen asignado el contenido, se realiza la misma operación para el texto completo, es decir, letra a letra se va comprobando el valor de cada una y el texto que más puntuación tenga, será el que más convenga al alumno. En el caso de ser manuales, todos los modelos se presentarán al usuario para que elija entre ellos en base a su criterio y al valor, que se mostrará para poder tener un mejor criterio a la hora de elegirlos.

3.6. Modelos de generación automática de ejercicios

El sistema realiza la generación de ejercicios de forma automática dependiendo de las necesidades del alumno. Si bien una forma totalmente adaptativa puede ofrecer

la potencia requerida, son diversos los motivos por los que el educador puede desear que los ejercicios se realicen de una forma más estricta. Puede querer que todos los alumnos realicen una serie de pasos (para todos igual) para comparar el nivel que tiene cada uno de ellos, o bien porque considere que son los más adecuados y no desea que se generen con otro criterio, o bien porque quiera relacionar los ejercicios de escritura con otro tipo de material didáctico (por ejemplo, que los ejercicios hablen de animales si es eso lo que se está enseñando a los alumnos en ese momento). Existen multitud de razones, ya que la enseñanza no es una ciencia exacta como bien nos enseña el refranero español con el dicho "cada maestrillo tiene su librillo". Por tanto nos tenemos que adaptar a esa situación y ofrecer al educador todas las posibilidades.

En todos los casos se sigue un flujo de trabajo, es decir, una serie de pasos consecutivos. La diferencia entre los modelos estriba en la forma de definir estos pasos.

3.6.1. Modelo fijo

En este modelo cada paso está asociado a unos ejercicios concretos. Se puede cambiar el número de ejercicios que ha de realizar el alumno correctamente antes de progresar al siguiente paso, los umbrales que debe superar, etc. Pero siempre con los ejercicios que haya decidido el docente que se han de realizar.

Para llevar a cabo este modo se debe asignar los metadatos correspondientes a cada parte. Si se asocia al paso un metadato concreto y el mismo a uno o varios modelos, serán esos los que se generen para ese paso de forma unívoca.

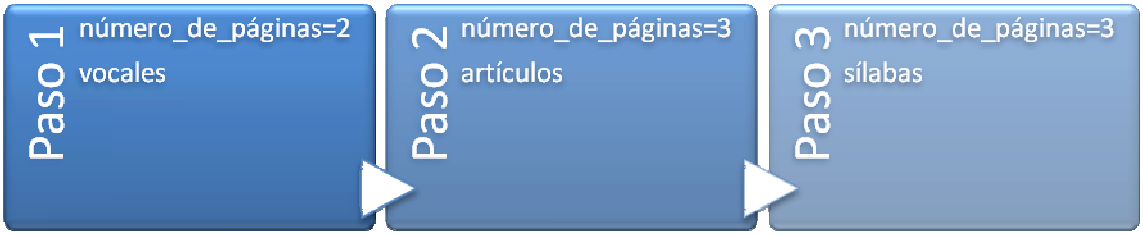

Figura 22: Ejemplo de pasos de Workflow en modelo fijo

Figura 23: Ejemplo de ejercicios para modelo fijo

En la Figura 22 se pueden observar los metadatos que deben tener los pasos para elegir los modelos de la Figura 23. El 'Paso 1' tiene un metadato 'vocales', por lo que sólo se podrán elegir para ese paso aquellos modelos que tengan dicho metadato. El único ejercicio que cumple dicha condición es el 'Ejercicio 1', por lo que es el que se asignará al alumno. Al pasar al 'Paso 2' observamos que el metadato 'artículos' sólo está asociado al 'Ejercicio 2', por lo tanto es el único que se podrá escoger cuando los alumnos se encuentren en dicho paso. Para el último paso el proceso sería exactamente el mismo.

3.6.2. Modelo flexible

El modelo flexible se diferencia del modelo fijo en que a cada paso se le asocia uno o una serie de plantillas predefinidas. Sin embargo el contenido se obtiene a partir del metadato correspondiente en el paso, que define una expresión regular de donde obtener las palabras válidas para esa plantilla. La elección de entre las candidatas vendrá dada por los valores que tenga el alumno en sus ejercicios anteriores. Se seleccionarán aquellas que puedan hacer hincapié en las letras que el alumno tenga más dificultad en realizar.

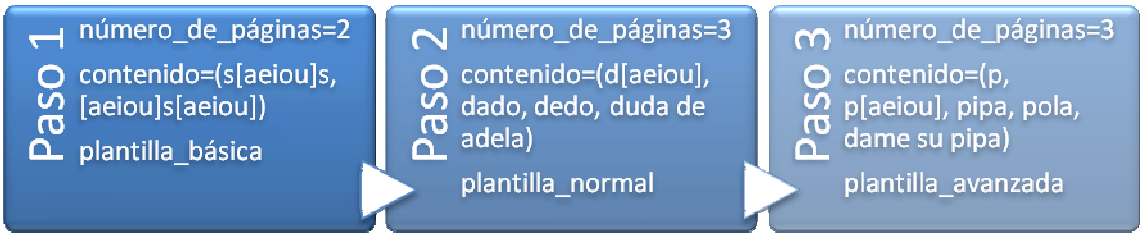

Figura 24: Ejemplo de pasos de Workflow en modelo flexible

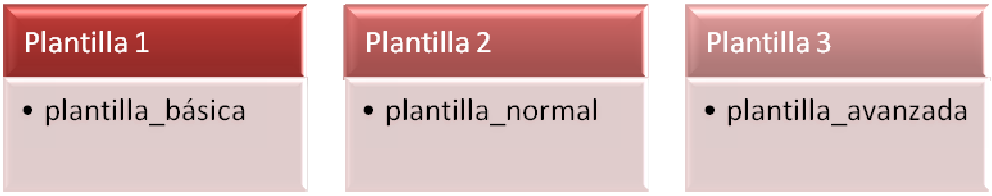

Figura 25: Ejemplo de plantillas para el modelo flexible

Como vemos en el 'Paso 1', sólo se seleccionarán aquellas plantillas que tengan el metadato 'plantilla_básica'. En este caso sólo existe la 'Plantilla 1' con ese metadato, por lo que será la que se escogerá en el 'Paso 1'. Si quisiéramos que en el 'Paso 1' se pudieran elegir varias plantillas bastaría con asociar el metadato 'plantilla_básica' a aquellas que considerásemos oportunas para realizar en dicho paso. Una vez escogida la plantilla, el contenido que se le puede insertar es el definido por el metadato

'contenido'. Para ello se utilizan expresiones regulares de Java. Por ejemplo, la cadena "s[aeiou]s, [aeiou]s[aeiou]" genera todas las palabras de tres letras que empiezan y terminan por 's' con una vocal en medio o bien empiezan y terminan por vocal y tienen una 's' en medio.

3.6.3. Modelo adaptativo

Por último está el modelo adaptativo, en el cual se elige la plantilla y el contenido en base a los metadatos definidos en cada elemento y a los datos del modelo de usuario. En este modelo es donde se aprovecha todo el potencial de los metadatos. El problema viene en que mientras más flexible sea el modelo, más complejo se vuelve. Por tanto habrá que definir más metadatos y estudiar mejor cada caso para que se generen los ejercicios deseados.

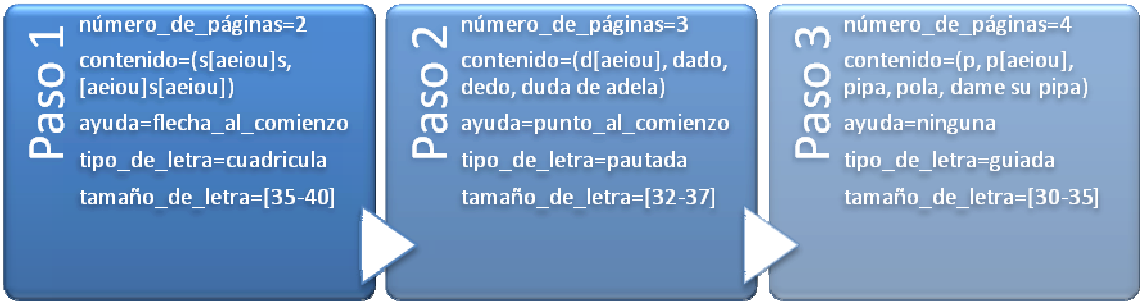

Figura 26: Ejemplo de pasos de Workflow en el modelo adaptativo

En la Figura 26 se muestra un ejemplo con tres pasos de workflow en los que se consideran los siguientes metadatos:

- **Número de páginas**. Valor numérico que indica el número mínimo de ejercicios que debe realizar el alumno antes de pasar al siguiente paso. El propósito es que un alumno no pase al siguiente paso porque haya realizado un ejercicio correctamente por casualidad. Mientras más importante sea el paso, mayor número de páginas pondremos ya que nos tenemos que asegurar que el alumno ha asimilado correctamente el aprendizaje.
- **Contenido**. Es una expresión regular que indica las posibles palabras que se pueden utilizar en la generación de un ejercicio. El sistema asignará un valor "educativo" a cada una de ellas dependiendo del modelo de usuario del alumno y elegirá las más convenientes.
- **Ayuda**: Indica si debe utilizarse la fuente con determinadas marcas para ayudar al alumno. Hay varios tipos de marca. Una flecha donde debe comenzar el alumno a escribir y el sentido en el que lo debe hacer. También puede ser simplemente un punto que indica por dónde debe iniciar.

- **Tipo de letra**: Indica el tipo de letra que se va a utilizar. Dependiendo del momento del aprendizaje en el que se encuentre el alumno, es conveniente que tenga ayudas que faciliten la tarea, como puede ser una cuadrícula o unas líneas que indiquen el tamaño de la letra. Conforme el alumno va adquiriendo la habilidad necesaria, se le van quitando estas ayudas.
- **Tamaño de letra**: Este parámetro indica el tamaño de la fuente que puede tener el ejercicio en un paso determinado. Generalmente se especificará un intervalo. Al principio la letra podrá ser más grande, pero poco a poco la iremos empequeñeciendo para que el alumno siga aprendiendo.

En ese ejemplo puede observase que entre cada paso existe variación en los valores de los metadatos, por ejemplo, el valor del campo contenido entre los pasos 1, 2 y 3 o el tipo de letra recomendado en cada uno de ellos. Este mecanismo de metadocumentación permitirá al algoritmo adaptativo y de generación automática de ejercicios (que se describe en el apartado 3.5 de esta memoria) escoger o crear los ejercicios que optimicen el valor educativo de cada ejercicio.

Como se puede observar, la cantidad de metadatos necesaria aumenta considerablemente. Hay que tener en cuenta que el ejemplo es sencillo para que el lector pueda comprender los conceptos sin que deba realizar un esfuerzo especial. Si se quiere diseñar un Workflow realmente adaptativo y complejo habría que añadir algunos más.

3.7. La interfaz de usuario

Esta interfaz es una herramienta genérica e independiente de esta aplicación que fue diseñada e implementada enteramente por el autor de este PFC. Tiene tres modalidades principales:

- Ventanas internas
- Pestañas
- Ventanas independientes

Se puede cambiar entre ellas conservando todas las ventanas abiertas. La más común es la interfaz de ventanas internas. En todas ellas existe una barra de herramientas con los diferentes iconos que abren las distintas partes del sistema, como puede ser el generador de plantillas, el de ejercicios, etc. Un ejemplo de la interfaz en modo de ventanas internas se muestra en la Figura 27.

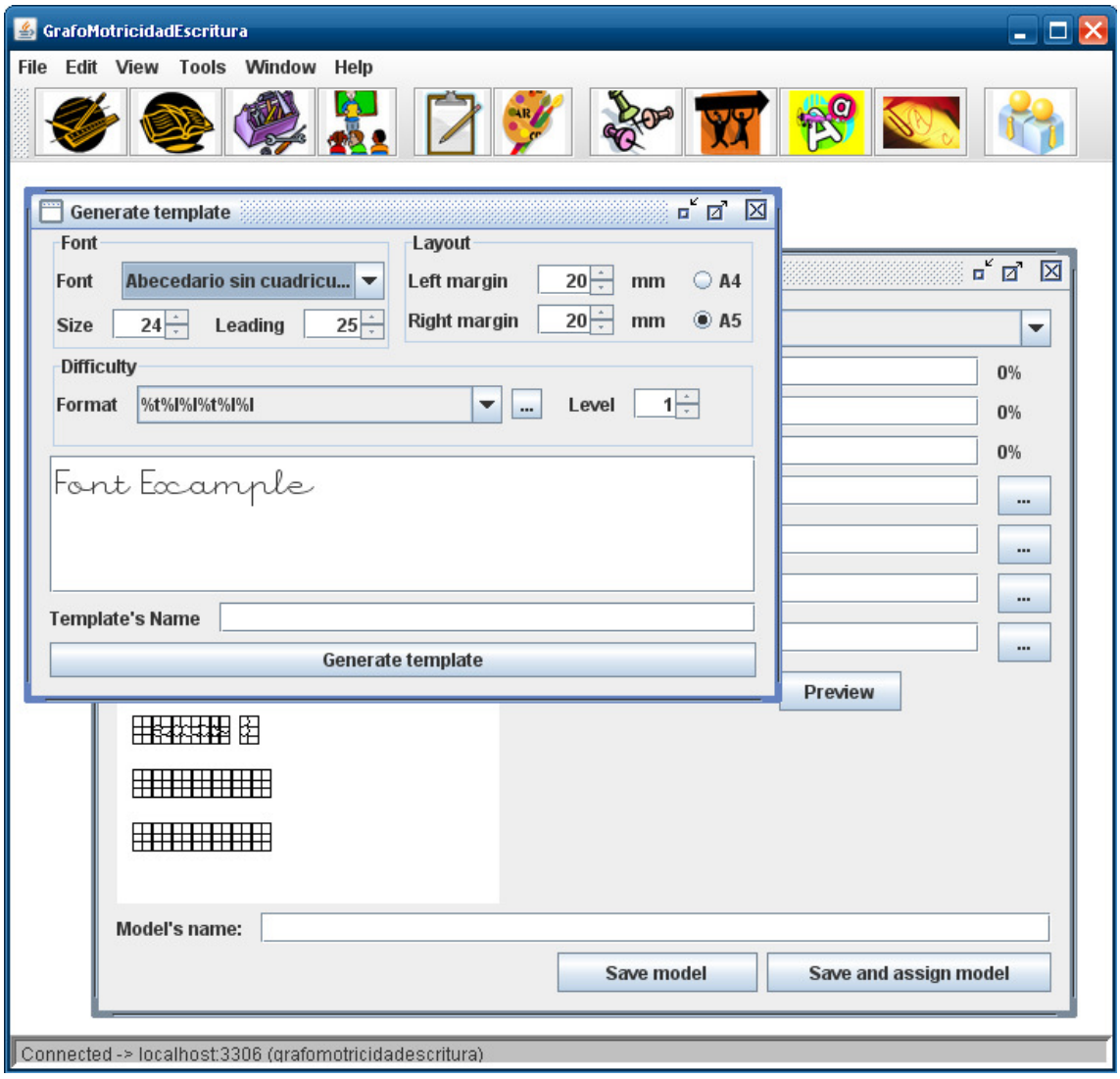

Figura 27: Ventana principal de la interfaz

En la parte alta se encuentran los diferentes botones. Estos botones estarán habilitados o no dependiendo del perfil de usuario y de sus privilegios en el sistema. En este modo existe un espacio donde se van a abrir todas las ventanas, las cuales las podemos maximizar, minimizar, cerrar, etc.

En la parte inferior existe una barra de estado en la cual se muestran diferentes mensajes.

En las siguientes secciones se describen cuáles son y para qué sirven cada una de las ventanas del sistema.

3.7.1. Login

El login es la primera ventana que nos aparece al ejecutar el programa. En ella debemos autenticarnos y dependiendo de la persona que sea, se le darán unos privilegios u otros.

Existen dos tipos de login: el de usuarios y el de estudiantes. Para elegir el tipo, basta seleccionar el "radio button" correspondiente, que son los dos círculos superiores que se verán en los siguientes ejemplos.

El login de usuarios se muestra en la Figura 28. En él podemos observar cómo hay dos espacios para poder insertar el usuario y el password. Una vez insertados basta con pulsar el botón "Aceptar". Si se pulsa el botón "Cancelar" no se autenticará ningún usuario y por tanto no se podrá acceder a las diferentes partes del programa, aunque una vez abierto, nos podremos autenticar nuevamente.

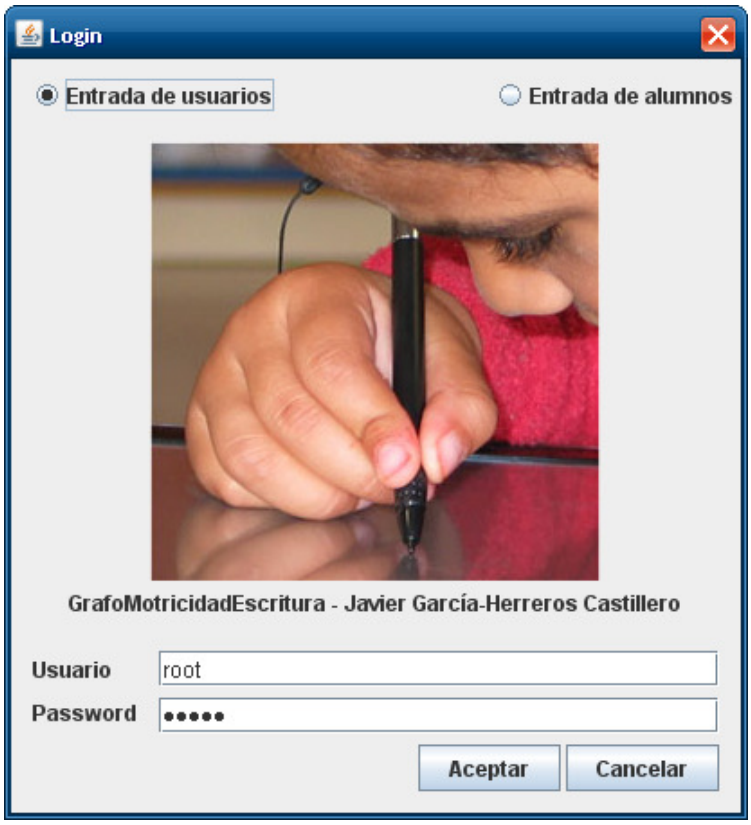

Figura 28: Login de usuarios

A continuación, en la Figura 29, se muestra el login de alumnos. Como se puede observar no se basa en usuarios y passwords, sino en simples botones. A los alumnos que tengan asociada una fotografía (véase el apartado 3.7.10) se les mostrará para que se puedan identificar. Si por cualquier motivo el alumno no tiene una foto asociada, se mostrará el nombre. Este sistema de autenticación se ha mostrado efectivo con niños de 3 a 5 años, ya que en esa edad todavía no son capaces de recordar e insertar un usuario y password en el sistema. El sistema probado, evidentemente tenía fotografías reales y no sintéticas como en el ejemplo.

Figura 29: Login de alumnos

3.7.2. Diseñar plantilla

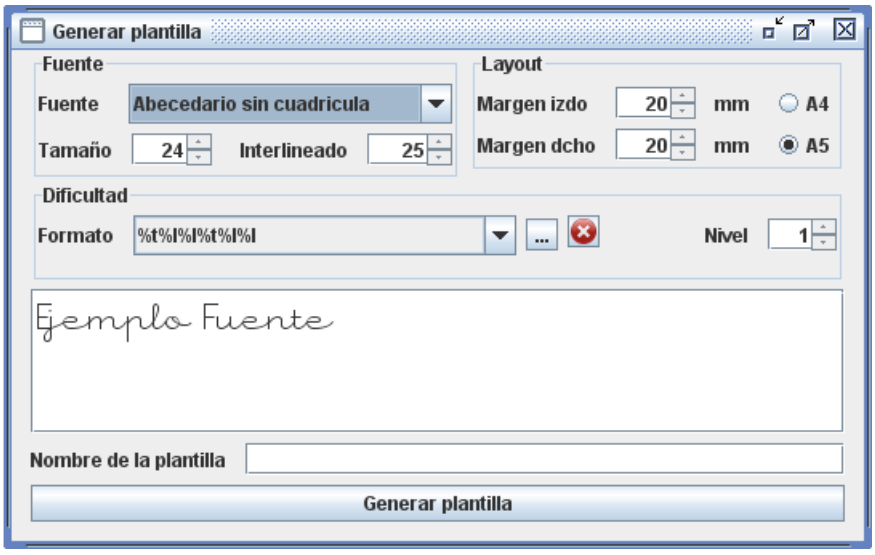

Figura 30: Ventana para generar las plantillas

Esta ventana se utiliza para generar plantillas para los ejercicios. Las distintas opciones son:

- **Fuente**: Define la fuente que se va a utilizar en la plantilla
- **Tamaño**: Define el tamaño de la fuente.
- **Interlineado**: Se utiliza para especificar la distancia entre una línea de texto y la siguiente.
- **Margen izdo/dcho**: Define los márgenes del texto con respecto a los bordes.

- **A4/A5**: Se utiliza para escoger entre papel a tamaño ISO A4 o A5.
- **Formato**: Especifica cuántas líneas de patrón va a haber, cuántas para que escriba el alumno y cuántos dibujos. %t especifica que es una línea de texto, es decir, el patrón para que lo copie el alumno. %l es una línea donde tendrá que escribir el alumno y %d especifica que va a haber un dibujo.
- **Nivel**: Define el nivel de dificultad de la plantilla.
- **Nombre de la plantilla**: Cada plantilla lleva asociado un nombre, el cual se va a utilizar para hacer referencia a ella. Habrá que escribir en la caja de texto el nombre deseado antes de generar la plantilla.

Una vez definidos todos los parámetros sólo queda pulsar el botón inferior "Generar plantilla" para realizar dicha acción, lo cual hace que se guarde la información en la base de datos y esté disponible en el resto de partes del sistema en las que se trabaja con plantillas, como en la ventana para diseñar el ejercicio que veremos a continuación.

Para crear un nuevo formato, basta con pulsar el botón '…', insertar el formato en el nuevo diálogo que saldrá y pulsar sobre el botón 'Aceptar'. Para eliminar uno existente, simplemente se pulsa sobre el botón rojo con la 'X' interior blanca y borrará el que en ese momento se encuentre seleccionado. Si el formato se está utilizando en alguna plantilla existente, el sistema mostrará un mensaje de error indicando que no es posible realizar la acción.

3.7.3. Diseñar ejercicio

Una vez diseñado una plantilla pasamos a diseñar un ejercicio, es decir, escoger el texto de cada línea de patrón y las imágenes a mostrar. En la **¡Error! No se encuentra el origen de la referencia.** se muestra una ventana de este tipo, donde se puede ver que a la izquierda aparece un ejemplo de la plantilla elegida y a la derecha la zona donde hay que insertar los distintos datos.

Dependiendo del número de líneas de texto y de dibujos que haya en la plantilla, aparecerán en la parte derecha los correspondientes elementos para rellenarlos. Para las líneas de texto basta con escribir en la caja de texto correspondiente. Mientras se esté escribiendo el indicador de línea completada se irá actualizando, mostrando en cada momento qué porcentaje del ancho de la línea se está ocupando. En ningún momento podrá pasar del 100% (se mostrará un mensaje en ese caso y no dejará escribir más). Puede ocurrir que no haya llegado al 100% pero no permita escribir más. Esto es debido a que esa letra sobrepasaría el ancho especificado. Para saber cuánto se puede escribir se tiene en cuenta el tamaño de la fuente, el tamaño del papel elegido (A4 ó A5) y los márgenes escogidos.

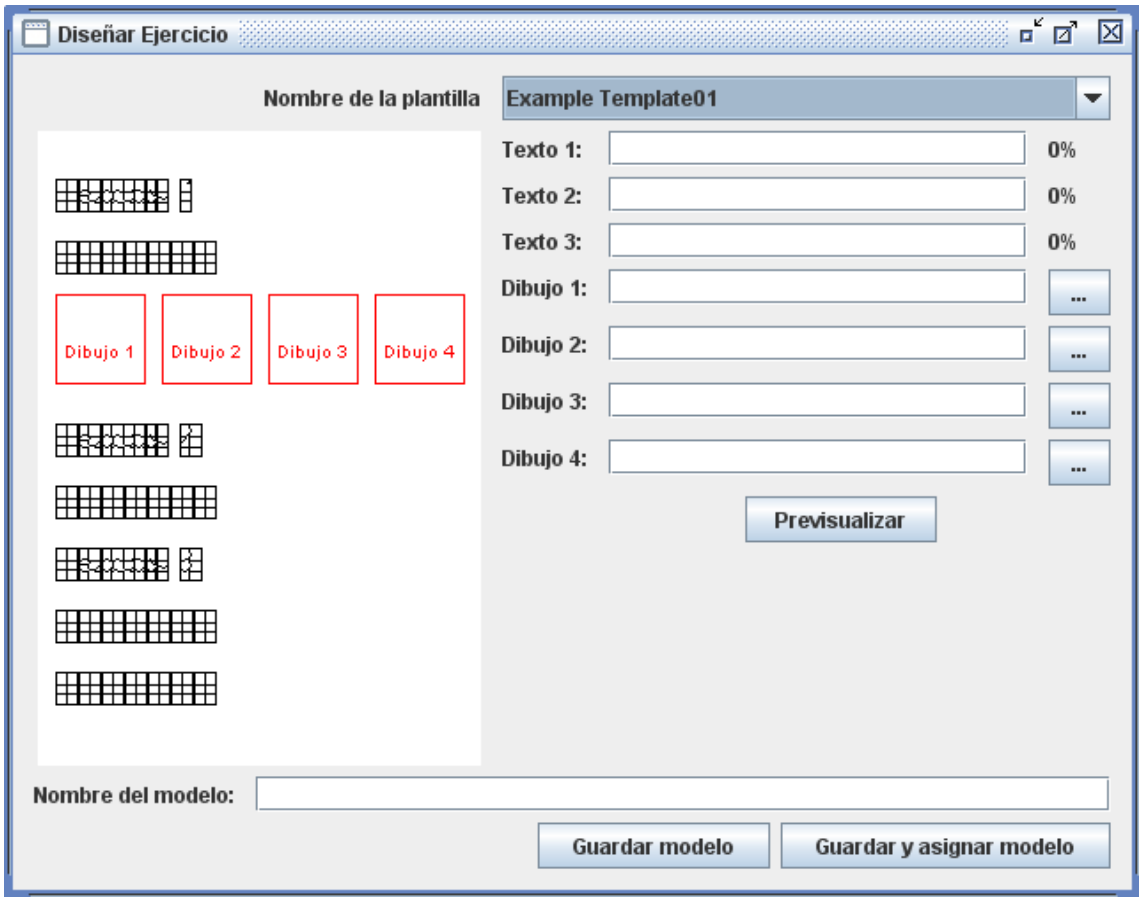

Figura 31: Ventana para generar los ejercicios

Para seleccionar las imágenes se inserta la dirección de la imagen en el disco duro en las cajas de texto o bien se seleccionan con un explorador de archivos que se abrirá al pulsar sobre el botón a la derecha de cada caja de texto habilitada para tal fin.

Al terminar de especificar los parámetros hay que seleccionar un nombre para el modelo. Entonces se podrá pulsar sobre el botón “Guardar modelo” o sobre “Guardar y asignar modelo”. Si se ha dejado algo por completar se avisará al usuario antes de generar el modelo.

Al pulsar sobre “Guardar y asignar modelo” no sólo se guardará sino que también se abrirá la ventana (o se seleccionará en caso de que esté abierta) para poder asignar el ejercicio a un alumno.

Si todo es correcto el sistema realizará los respectivos cambios en la base de datos y avisará al usuario con un mensaje de que la acción se ha realizado correctamente.

3.7.4. Asignar modelos

La ventana de la Figura 32 se utiliza para asignar los modelos (ejercicios) a los alumnos. La asignación se puede realizar de forma manual o automática.

Para asignarlos manualmente, en el árbol de la izquierda se ordenan los diferentes modelos por nivel y por formato. Una vez elegidos los distintos modelos

(pueden ser más de uno a la vez) se seleccionan los alumnos a los que se van a asignar. Si se van a imprimir para que los realicen en papel habrá que marcar la casilla "Imprimir modelo(s)".

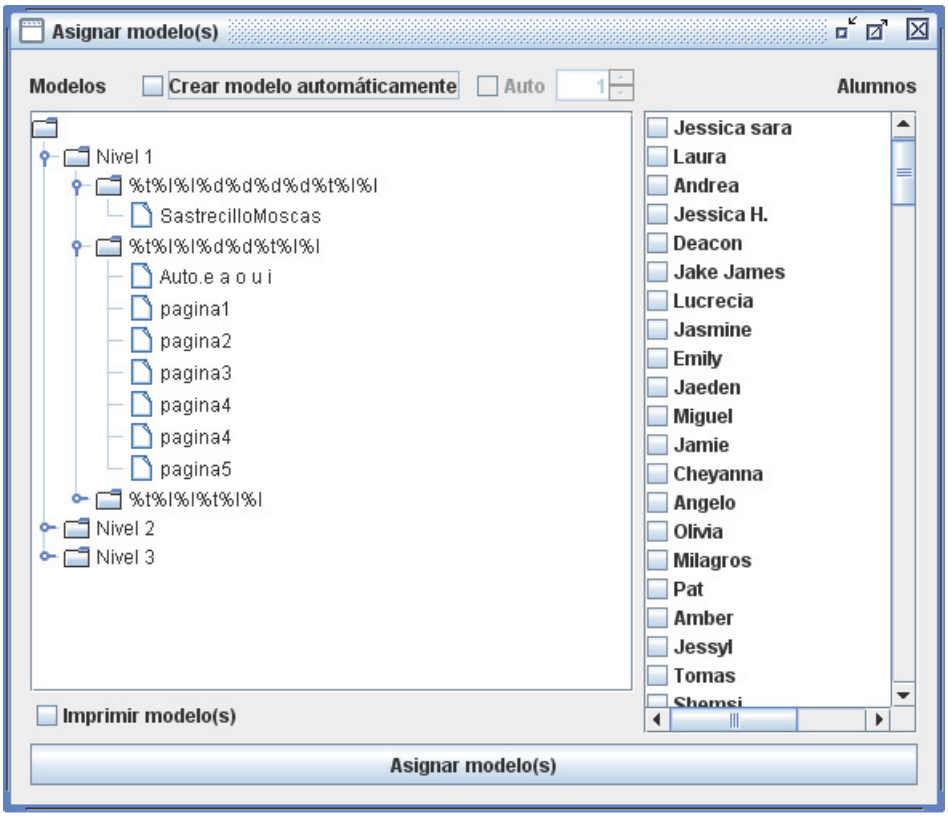

Figura 32: Ventana para asignar los modelos

Para asignar de forma automática los modelos basta con activar la casilla 'Crear modelo automáticamente'. En ese momento se habilitará la casilla 'Auto'. Esta casilla se utiliza para que el sistema no pregunte al usuario qué modelos quiere asignar de entre los disponibles, sino que se realiza una valoración de cada uno de ellos y todo el proceso se realiza sin ninguna interrupción. Si se activa esta opción, se habilitará seguidamente el spinner a la derecha. Con él, se pueden seleccionar cuántos ejercicios por alumno se quieren crear. Si no se activa la casilla 'Auto', el sistema ofrecerá al usuario la posibilidad de elegir los ejercicios entre los que se han valorado a partir de los metadatos tal y como se muestra en la Figura 33.

Una vez elegidos los alumnos y los ejercicios (ya sea de forma manual o automática), se pulsará el botón inferior y esto hará que se generen automáticamente (se creará un archivo con el ejercicio y otro con la solución por cada alumno, con su código de barras específico, etc.) y, en caso de haberlo seleccionado, se imprimirán para poder dárselo a los alumnos para que los completen.

Si los ejercicios no se imprimen, estarán disponible en el sistema para hacerlo posteriormente (véase apartado 3.7.8) o para que los completen con el tabletPC.

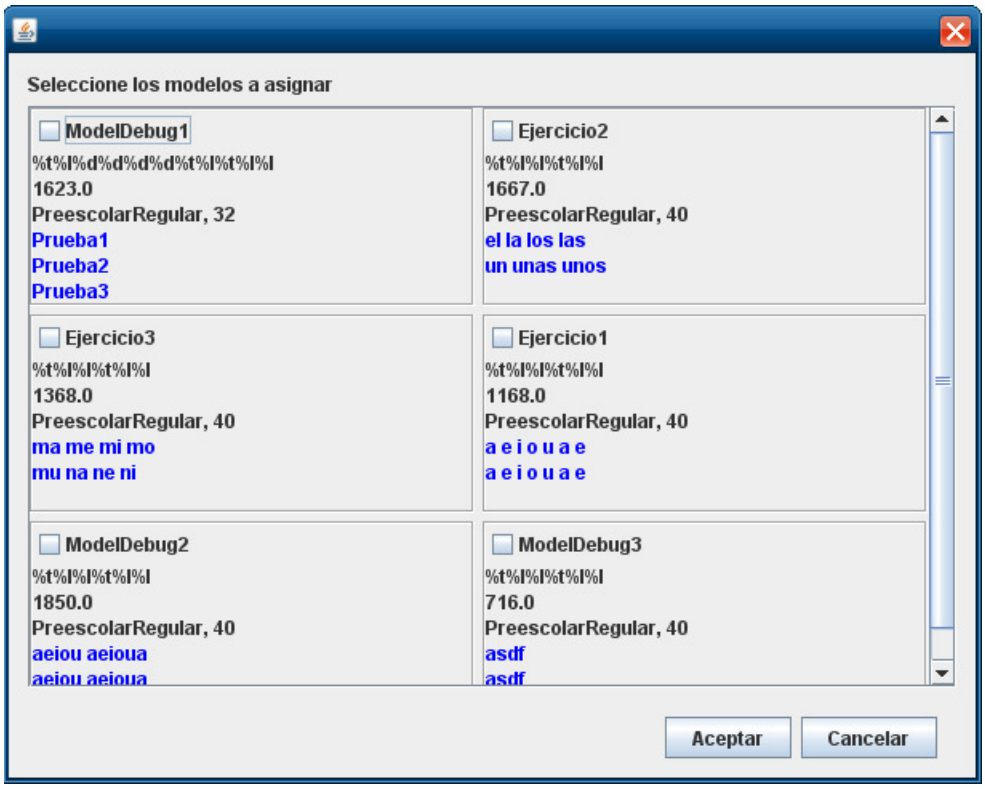

Figura 33: Selección de los modelos propuestos por el sistema

3.7.5. Gestión de metadatos

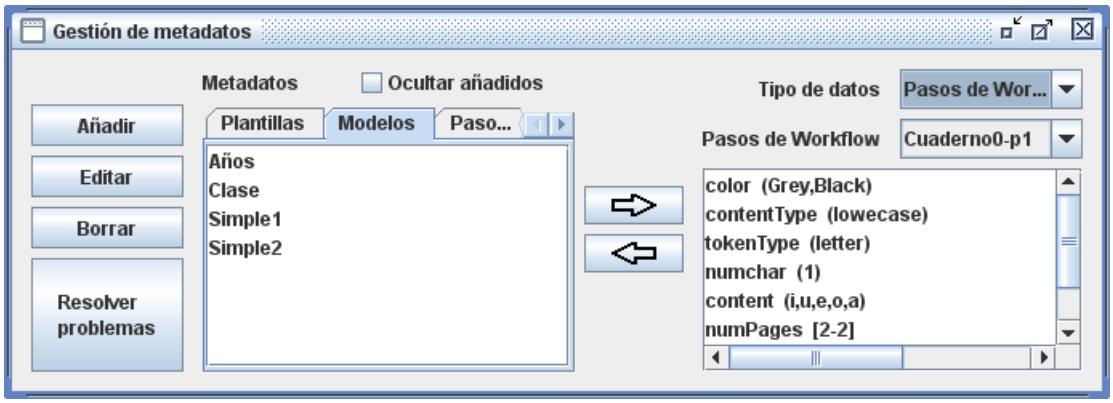

Figura 34: Ventana para gestionar los metadatos

Esta ventana se utiliza para gestionar los diferentes metadatos de las plantillas, los modelos y los pasos de workflow. A la izquierda existen cuatro botones para añadir, editar y borrar metadatos y un cuarto botón para resolver problemas que veremos a continuación.

Los metadatos se pueden clasificar por el tipo de dato al que irá asociado (plantillas, modelos o pasos de workflow) o bien por el tipo de metadato en sí (simple, conjunto de valores o intervalo de enteros).

A un paso de workflow se le puede asociar cualquier tipo de metadato. Si se le asocia un metadato de plantillas, se utilizará para comparar éstas a la hora de generar automáticamente los ejercicios. Ocurre lo mismo con los metadatos de modelos. Asignados a un paso de workflow son útiles a la hora de saber qué modelo escoger.

A una plantilla sólo se le pueden asociar metadatos de plantillas, al igual que a los modelos, que sólo se le asignaran metadatos de su tipo.

Cuando se añade un nuevo metadato, ya sea de modelos, de plantillas o de pasos de workflow, hay que especificar el nombre y el tipo, que puede ser 'simple', 'conjunto de valores' o 'intervalo de enteros'. Un metadato simple es aquel al que se le da sentido por el mero hecho de estar asociado a un elemento. Sin embargo los demás tipos, además de estar asociados a un elemento deben tener cierto valor. Por ejemplo, si el tipo es 'conjunto de valores' se ha de especificar qué valores va a tomar el elemento. Un ejemplo de esto sería el metadato 'Color'. Este metadato puede tener asociados ciertos valores, como puede ser 'Rojo, Azul, Verde, Naranja, Amarillo'. Pero quizás sólo se quiera asociar uno de ellos a un elemento determinado. Para ello primero hay que asignarle los posibles valores que puede tomar el metadato pulsando con el botón derecho sobre el metadato deseado y eligiendo la entrada del menú "Editar posibles valores". En caso de ser un tipo 'conjunto de valores' habrá que especificar los valores separados por comas. Si es un 'intervalo de enteros' simplemente bastará con seleccionar el mínimo y el máximo del intervalo.

Si se asigna un metadato a un elemento pero el metadato todavía no tiene especificados los valores que puede tomar se avisará al usuario de tal hecho y aparecerá una ventana para hacerlo. Una vez especificados se podrán seleccionar los deseados.

Al crear un metadato, éste se asociará a un dato u otro dependiendo de la pestaña que se tenga seleccionada. Si es la de plantillas, por ejemplo, y se crea un nuevo metadato, éste sólo se podrá asociar a plantillas o a un paso de workflow. Sólo se mostrará en la lista de metadatos cuando se tenga seleccionada la pestaña correspondiente.

Para asignar un metadato a un elemento primero hay que elegir éste. Para ello se elige en el combo "Tipo de Datos" el tipo deseado (ya sean plantillas, pasos de workflow, etc) y a continuación se podrá escoger en el combo inferior el elemento concreto. Una vez elegido el elemento ya podemos seleccionar un metadato y pulsar sobre el botón "->" para añadirlo. En caso de no ser un tipo simple aparecerá una ventana para asignar los valores deseados.

Hay que darse cuenta que si se selecciona una plantilla o un modelo, sólo estarán disponibles las pestañas de su tipo. Sin embargo, al elegir un paso de workflow estarán disponibles las tres pestañas. Esto es así debido a lo comentado al comienzo del apartado, es decir, qué metadato se puede asociar a qué elemento.

Para borrar un metadato de un elemento basta con seleccionarlo en la ventana derecha y pulsar sobre el botón "<-".

El botón "Resolver problemas" viene dado por la problemática que se puede crear al modificar los posibles valores de un metadato. Supongamos que existe un metadato 'Color' con los valores 'Azul, Rojo, Verde'. Si tenemos un elemento con el color 'Verde' y cambiamos los posibles valores del metadato a 'Azul, Rojo', esto hará que el elemento que tenía asociado el valor 'Verde' ya no lo tenga, con lo que tiene un metadato que no es simple sin valores. Al pulsar sobre este botón se visualizarán todos los elementos que tienen este tipo de problemas y podremos editarlos o borrarlos como se puede ver en la Figura 35.

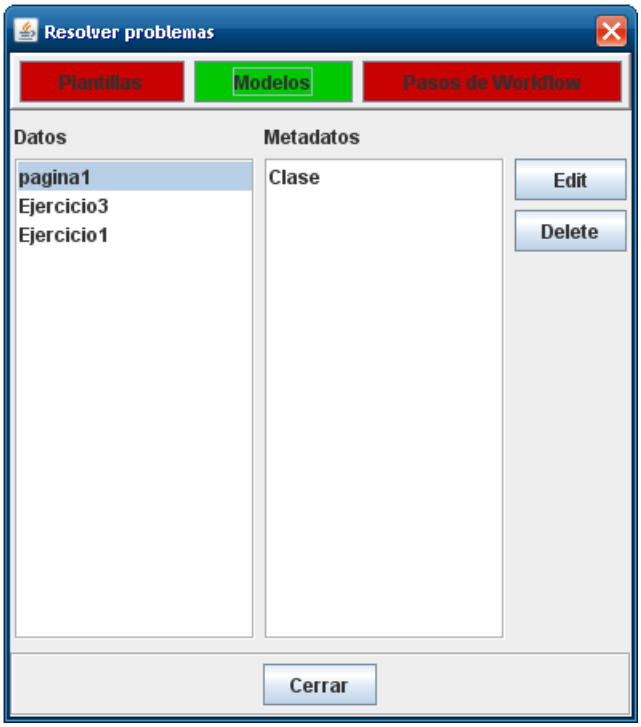

Figura 35: Ventana de resolución de problemas de metadatos

En la parte superior tenemos los diferentes elementos que pueden contener metadatos. Cuando se selecciona uno de ellos aparece en la lista 'Datos' aquellos elementos que tienen algún metadato vacío. Al seleccionar uno de los metadatos problemáticos podremos editar su valor pulsando sobre el botón 'Editar' o bien borrarlo pulsando sobre el botón 'Borrar'

3.7.6. Gestión de workflow

El proceso de aprendizaje de la escritura se realiza mediante un sistema de fases. En el caso de nuestro sistema, estas se pueden realizar de forma manual (gestionadas totalmente por el profesor), semi-automática (el profesor elige la fase y el sistema sugiere el ejercicio) o completamente autónoma (el sistema asigna y propone las tareas a realizar). En cada fase se van introduciendo nuevas letras, se modifica el tipo

(cuadriculada, pautada, etc.), se incrementa la dificultad, etc. Al conjunto de esos pasos a seguir nosotros lo llamamos workflow, ya que son una serie de trabajos que hay que realizar en serie y con etapas bien definidas. Cada una de estas etapas la llamamos "paso de workflow". Puede haber diferentes tipos de workflow, unos que vayan más rápido en el aprendizaje y cuya dificultad aumente rápidamente y otros centrados en alumnos con necesidades especiales que puedan tener otro tipo de dificultad, cambiando las letras, etc. Con la ventana de la Figura 36, se pretenden gestionar todos ellos, es decir, crearlos, encadenarlos, modificarlos, etc.

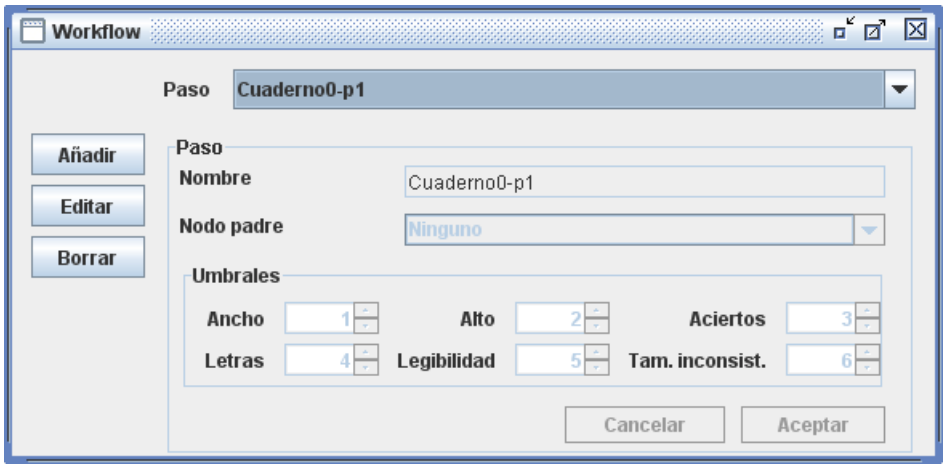

Figura 36: Gestión del workflow

Mediante esta ventana se pueden añadir pasos, editarlos y borrarlos. La idea es que cada workflow tiene un nodo raíz que es por el que se empezará. Por tanto este nodo raíz será el único tipo que no tenga nodo padre. Todos los demás tendrán como nodo padre el paso que le preceda.

Además, cada paso de workflow tiene asignado unos umbrales, que son los que el alumno deberá superar para avanzar al siguiente paso.

- El **Ancho** especifica si el alumno hace las palabras de la misma longitud que las originales. Tanto si las hace más anchas o más estrechas, este valor bajará a la hora de corregir el ejercicio.
- El **Alto** es similar al Ancho pero comprueba la altura de las letras en vez de la anchura.
- Los **Aciertos** especifica el porcentaje de lo escrito por el alumno que se encuentra en una posición correcta.
- El dato **Letras** detalla el porcentaje de cada letra que ha realizado correctamente el alumno.
- La **Legibilidad** pretende dar una idea de lo bien que realiza las letras el alumno en general.

- El **Tam. Inconsist.** (Tamaño inconsistente de letra) se utiliza para ver si el alumno varía el tamaño de las letras mientras está escribiendo. Un valor de 100 significa que todas las letras son del mismo tamaño, mientras que un valor de 0 indica lo contrario.

Para añadir un nuevo paso basta con pulsar el botón "Añadir" y rellenar el campo "Nombre" y "Nodo padre". Una vez realizado este paso, se pulsará sobre "Aceptar".

Si se pulsa sobre "Editar" se habilitarán los campos "Nombre", "Nodo padre" y los umbrales para poder modificarlos. Si se pulsa sobre "Aceptar" se guardarán los cambios. En cambio, si se pulsa sobre aceptar los datos quedarán inalterados.

Para borrar un paso simplemente se selecciona el paso deseado mediante el combo "Paso" y a continuación se pulsa el botón "Borrar". En caso de que el paso sea padre de algún otro paso, se mostrará un mensaje de error y no se realizará la acción. Esto quiere decir que sólo se podrán borrar los nodos hojas, es decir, aquellos que sean el último paso de un workflow.

3.7.7. Monitorización de estudiantes

La monitorización de estudiantes se utiliza para comprobar el progreso de los diferentes alumnos a lo largo de su aprendizaje. El progreso se visualiza mediante una gráfica. Los datos para mostrar la gráfica pueden ser filtrados tal y como se explica más adelante.

Existen tres modelos de gráficas: Categorías, gráfica en el tiempo o en barras.

- **Gráfica de categorías**: Muestra el resultado de cada ejercicio sin tener en cuenta la fecha en la que se ha resuelto. Es el ejemplo que se puede observar en la Figura 37, donde cada uno de los ejercicios realizados están equidistantes entre sí. Esta gráfica se utiliza para observar cómo ha afectado cada ejercicio a la habilidad del alumno.
- **Gráfica de barras**: Este tipo de gráfica está pensada para realizar comparaciones entre los alumnos. Se muestra una media de todas las características que se hayan señalado en un diagrama de barras junto con la media de los alumnos y aquellos que se señalen en la columna de la izquierda. De esta forma podemos comprobar si el alumno en cuestión va más retrasado que la clase, si su progreso es parecido al de la media o si va más adelantado.
- **Gráfica en el tiempo**: Se utiliza para comprobar el aprendizaje de un alumno en el tiempo ya que cada resultado se muestra según la fecha en la que se haya realizado el ejercicio. Así podemos observar cómo el

alumno progresa con respecto a su edad. La gráfica es igual que la de categorías. La única diferencia es que la posición del resultado en el eje de abscisas será directamente proporcional a la fecha y no serán equidistantes entre sí.

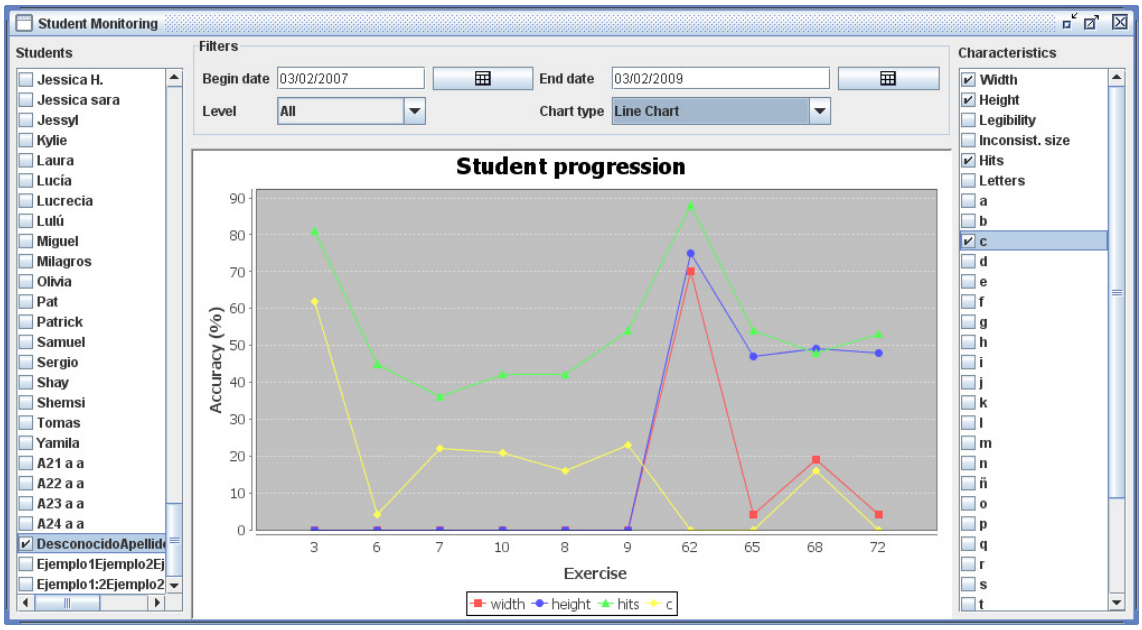

Figura 37: Monitorización de alumnos

Para mostrar los resultados tenemos que seleccionar 6 parámetros:

- **Los estudiantes**: En la columna de la izquierda se muestran todos los estudiantes ordenados por orden alfabético. Primero se ordena por el primer apellido, luego por el segundo y por último por nombre. Los alumnos a los que no se les haya insertado el apellido aparecerán al principio de la lista. Si la gráfica sólo muestra el resultado de un alumno y hay varios seleccionados, se escogerá el primero de ellos.
- **Fecha de inicio y fecha de fin**: Estas dos fechas seleccionan el intervalo de tiempo que queremos observar. Los datos que se muestre estarán dentro de estas dos fechas, independientemente de la gráfica que se elija. En el caso del diagrama de barras, la media se realizará con los datos de ese intervalo.
- **Nivel**: Filtra los resultados dependiendo del nivel del ejercicio. Esto puede resultar útil sobre todo a la hora de la comparación entre alumnos, para que no se mezclen los resultados de todos los niveles, ya que si la media está compuesta por todos los ejercicios que han realizado los demás alumnos pero el que queremos comparar sólo ha realizado ejercicios de nivel 1, el resultado de la comparación no es válida para sacar conclusiones.
- **Tipo de gráfica**: Escoge el tipo de gráfica a mostrar. Las diferencias entre las distintas gráficas las hemos comentado anteriormente.

- **Características**: En esta lista se seleccionan aquellos parámetros a mostrar de los que se evalúan.

3.7.8. Impresión de modelos

Al asignar los modelos puede que no se quieran imprimir en ese instante, por lo que existe una ventana para realizar dicha tarea. Se muestra en la Figura 38, la cual tiene tres zonas.

La primera es la zona de filtrado de estudiantes, donde se seleccionarán los alumnos para los que se quiere mostrar los ejercicios. Si se selecciona el checkbox "Todos", el filtro por estudiante no se tendrá en cuenta.

La segunda zona es la zona de filtros. Si no se quiere utilizar un filtro basta con marcar la casilla "Todos" para que no se tenga en cuenta. Existen cinco tipos:

- Filtro de **Plantillas**: En este filtro se muestran todas las plantillas disponibles. Si se selecciona una o varias, sólo se mostrarán los ejercicios que tengan dichas plantillas.
- Filtro por **nivel**: Este filtro hará que se muestren sólo los ejercicios de los niveles seleccionados, a menos que se marque la casilla "Todos", que en tal caso no se tendrán en cuenta y se mostrarán todos.
- Filtro por **fuente**: Si se selecciona este filtro hará que sólo se muestren los ejercicios que tengan una fuente de entre las seleccionadas.
- Filtro por **formato**: Hará que sólo se muestren los modelos cuya plantilla tenga un formato que se haya seleccionado.
- Filtro por **fecha**: Al activar este filtro, sólo se mostrarán aquellos modelos que han sido asignados entre las fechas especificadas. Las fechas se pueden insertar manualmente o bien pulsando en el botón asociado, el cual hará que aparezca un calendario en el cual seleccionaremos la fecha correspondiente.

Una vez que se hayan elegido todos los filtros que deseemos, habrá que pulsar sobre el botón "Actualizar" para que aparezcan los ejercicios que cumplan con las especificaciones de los filtros. En caso de que no haya ejercicios que cumplan las restricciones impuestas, no se mostrará ninguno y habrá que relajarlas (desactivar filtros o elegir más opciones). Cuando aparezcan los ejercicios hay que elegir los que se quieren imprimir. Para ello basta con hacer click con el botón izquierdo del ratón sobre ellos. El ejercicio pasará de color rojo a color verde, indicando que se va a imprimir. Para finalizar, basta con pulsar el botón "Imprimir" para que aparezca la ventana de impresión.

Como es normal, el ejercicio se muestra en un tamaño reducido para que quepan mejor en la ventana. En caso de querer ver uno en concreto a un tamaño mayor, basta con pulsar con el botón derecho sobre él y seleccionar la opción “Mostrar a tamaño completo”. Aparecerá una ventana con la imagen de ese ejercicio a la que podremos hacer zoom y movernos por ella con las barras de desplazamiento.

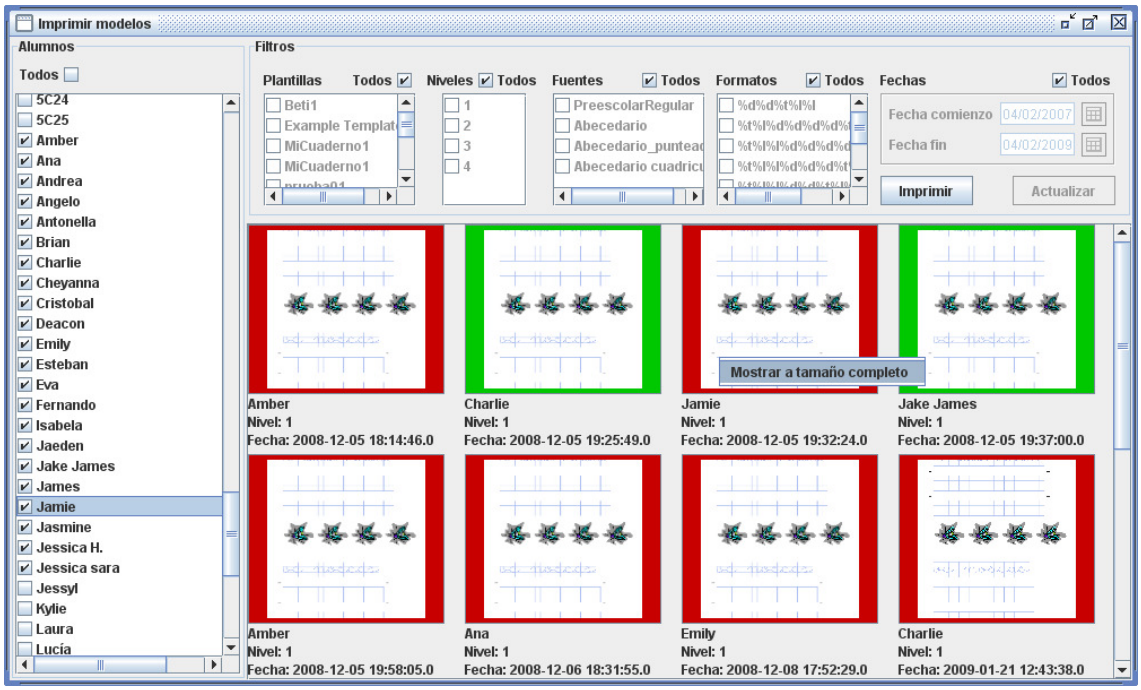

Figura 38: Impresión de modelos

3.7.9. Administración de usuarios

Para acceder al sistema hace falta autenticarse. Dependiendo de quién lo haga, tendrá unos permisos u otros. Estos permisos se establecen en la ventana que se muestra en la Figura 39.

Los permisos se pueden establecer de dos maneras: Individualmente o mediante perfiles.

Los perfiles son muy útiles a la hora de establecer permisos debido a que habrá muchos usuarios que los compartan, de forma que si se quiere cambiar los permisos de un grupo, basta con irse al perfil correspondiente y cambiarlos. De esta forma se cambiarán las de todos aquellos que tengan asignado dicho perfil y no habrá que ir uno por uno cambiándolos. Los grupos que pueden aparecer de forma natural en el sistema pueden ser profesores, alumnos, administradores, etc.

Como este sistema está diseñado para que pueda ser usado por niños muy jóvenes (más o menos desde 4/5 años) y es probable que no sepan escribir un usuario y contraseña, se ha establecido una autenticación especial basadas en fotos. A esas edades ya son capaces de reconocerse en una foto o leer su nombre (en caso de que

los padres no den permiso de sacarles foto y guardarlas en el sistema), por lo que simplemente se buscarán en una lista de fotos y se elegirán a ellos mismos. En general los usuarios que correspondan con un alumno sólo deberían tener habilitada la opción de hacer ejercicios, y escogerán entre los que tengan asignados y no realizados.

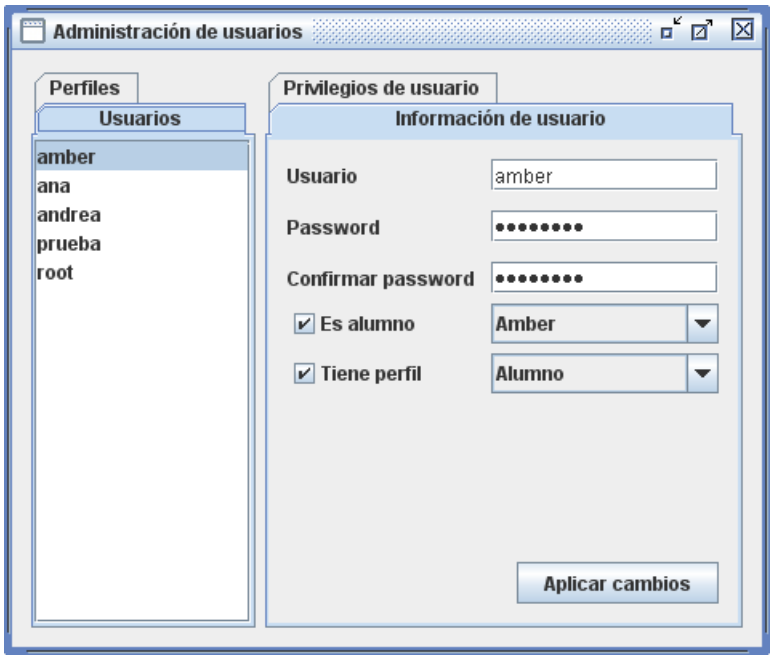

Figura 39: Administración de usuarios

Para crear un usuario basta con pulsar el botón derecho sobre la lista de usuarios y elegir la opción “Añadir usuario”. A continuación se rellenarán los datos del nombre de usuario, el password y se le asignará un alumno o un perfil en caso de que los tengan. Para que los cambios tengan efecto simplemente se pulsará el botón “Aplicar cambios”.

Para borrar un usuario se debe proceder de la misma forma que para añadirlo, solo que en vez de pulsar el menú “Añadir usuario”, se pulsará “Eliminar usuario”.

Si un usuario tiene asociado un perfil, no se le podrán cambiar los privilegios. Para ello se debe deshabilitar tal opción y asignarle privilegios individuales. En caso de que esté asociado a un perfil, si lo que se quiere hacer es cambiar los privilegios del grupo entero, se seleccionará la pestaña “perfiles”, se seleccionará el perfil que se desee cambiar, se modifican los permisos en la pestaña “Privilegios de usuario” (que se activará automáticamente al pulsar sobre la pestaña “perfiles”) y se pulsa sobre el botón “Aplicar cambios”.

Para añadir un perfil o borrarlo, se procede de la misma forma que para añadir o borrar un usuario, excepto que se hará teniendo la pestaña “Perfiles” seleccionada.

3.7.10. Gestión de alumnos

El apartado que nos ocupa permite gestionar los alumnos, cambiarles los datos personales, información del nivel en el que se encuentran, el perfil que se le quiera asignar, etc. Un ejemplo de la ventana se muestra en la Figura 40.

Para añadir un alumno basta con pulsar el botón “Añadir”, rellenar los campos correspondientes y pulsar “Aplicar cambios”.

Los campos **nombre**, **primer apellido**, **segundo apellido** se rellenan simplemente escribiendo la información en ellos. Para rellenar la **fecha de nacimiento** hay que seguir el formato dd/mm/yyyy y que la fecha sea correcta. De otra forma el sistema la corregirá. También se puede pulsar sobre el icono del calendario que aparece a la derecha del campo y elegir el día de nacimiento desde el mismo.

El campo **clase** especifica el aula en la que se encuentra inscrito el alumno. Se debe seleccionar la clase de entre las disponibles con la lista desplegable.

El campo **nivel** hace referencia al nivel del alumno. Se debe seleccionar de la misma forma que el campo clase. Este campo no se debería modificar a posteriori, ya que el alumno pasaría de clase automáticamente al realizar correctamente los ejercicios.

El campo **paso de workflow** establece el paso en el que se encuentra el alumno. Al igual que en el caso del nivel, sólo debería cambiarse para asignar el alumno al primer paso, ya que se irá cambiando automáticamente conforme vaya avanzando y el sistema corrija los ejercicios.

El campo **perfil** se utiliza para asignarle un perfil (que debería ser Alumno) al usuario asociado al alumno. En caso de no tenerlo, se creará automáticamente con el password por defecto, aunque no debería ser problema debido a que la autenticación de los alumnos no requiere contraseña. En caso de querer usar la contraseña, se debería ir a la ventana de gestión de usuarios (véase el apartado 3.7.9) y cambiarlo. Para ello hay que tener en cuenta que cuando se crea un usuario a un alumno automáticamente, el nombre del usuario es el nombre del alumno más las iniciales de sus apellidos. Por ejemplo, si el alumno se llamase Juan Fernández Sánchez, el nombre de su usuario sería “juanfs”. Si a la hora de crear un usuario automáticamente, ya existiera en la base de datos, se le pondría un índice. En el caso del ejemplo, el segundo usuario sería “juanfs1”, el tercero “juanfs2”, y así sucesivamente. En la ventana de gestión de usuarios aparecerá a qué alumno pertenece cada usuario.

La fotografía se utiliza para la autenticación basada en fotos. Para asignarle una fotografía a un alumno basta con pinchar sobre el recuadro de la misma, seleccionar el fichero correspondiente y pulsar sobre el botón “Aplicar cambios”. Si se desea borrar a

posteriori, habrá que pulsar sobre el botón “Eliminar fotografía” y a continuación a “Aplicar cambios”. Si un alumno no tiene asociada una fotografía, en la autenticación se mostrará su nombre.

Si lo que se desea es borrar un alumno, basta con seleccionarlo en la lista y pulsar el botón “Borrar”.

Existen determinados campos que se pueden asignar a varios alumnos a la vez. Estos son fecha de nacimiento, clase, nivel y paso de workflow. Esto se hace porque es probable que muchos de estos campos se cambien a la vez en varios alumnos. Como por ejemplo el paso de workflow. En principio todos los alumnos (o por lo menos los de la misma clase) deberían empezar en el mismo paso. También se puede querer asociar el perfil de Alumno a todos a la vez, o cambiar a todos los alumnos de una clase a otra en el cambio de curso, por ejemplo. Para ello se seleccionarán de la lista de alumnos todos aquellos a los que se le quiera cambiar los datos y a continuación se modificarían como si se tratase de un alumno individual.

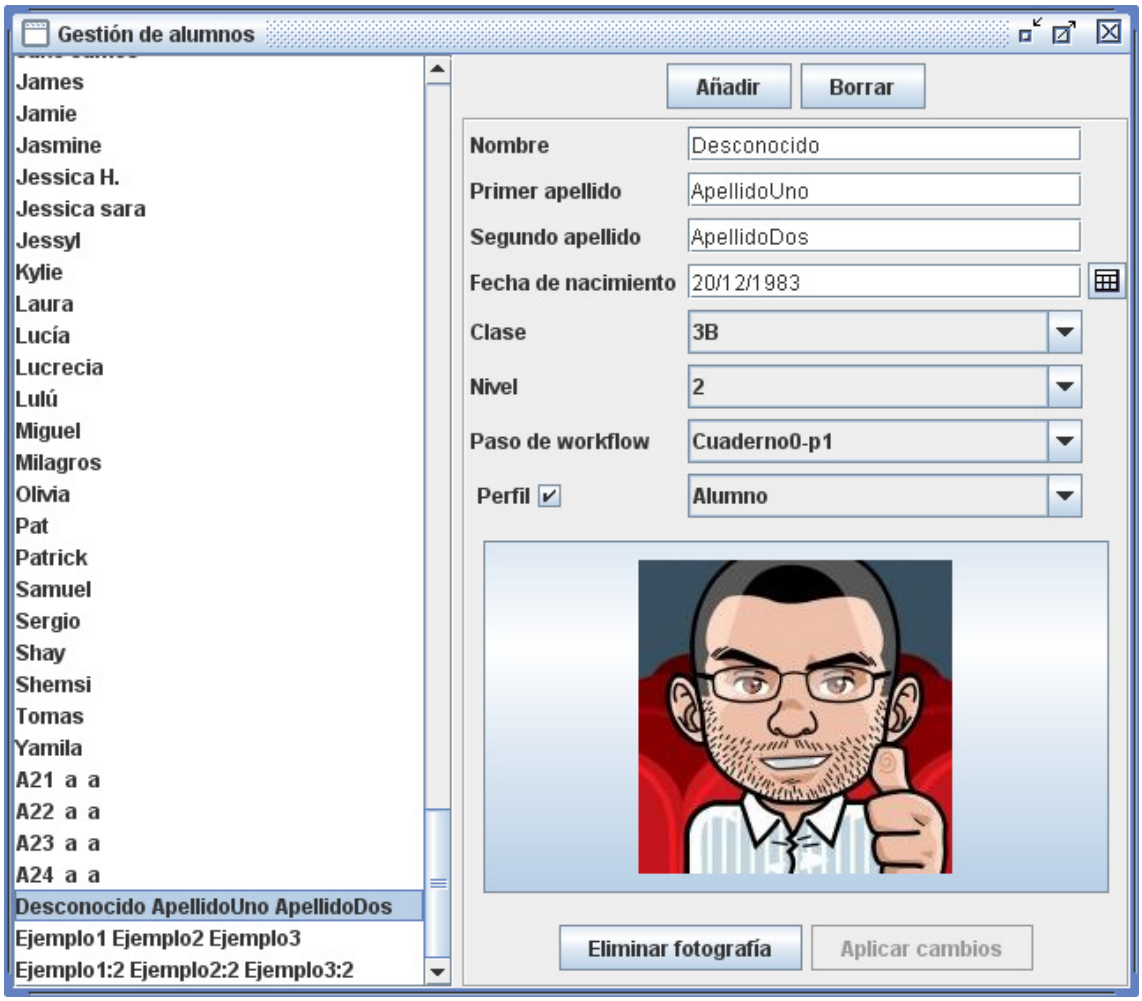

Figura 40: Gestión de alumnos

3.7.11. Corrección de ejercicios

Cuando se imprimen los ejercicios, a continuación se escanean para poder corregirlos. Para ello se utiliza la ventana de "corrección de ejercicios" mostrada en la Figura 41.

Lo primero que hay que hacer es seleccionar aquellos ejercicios escaneados que se quieran corregir. Basta con pulsar el botón "Seleccionar archivos" para añadir las imágenes que se deseen evaluar. Los seleccionados aparecerán en la lista como se muestra en el ejemplo. Si se ha seleccionado alguno por equivocación, basta con pinchar sobre ellos y pulsar la tecla "Supr".

Para comenzar la corrección automática de los modelos seleccionados, simplemente se pulsará el botón "Corregir ejercicios".

Existe una opción que es "Mostrar resultados", lo que hará que después de cada corrección se muestre una ventana con una imagen que superpondrá la solución óptima y la respuesta del alumno, para que podamos ver la diferencia. Además, la respuesta del alumno estará en color verde en aquellas zonas que se ha considerado correcta y rojo donde no lo haya realizado con precisión suficiente. También se mostrará a la derecha los diferentes valores de la corrección, como el ancho, alto, letras, etc. Se puede ver en la Figura 42.

Para una óptima corrección se recomienda utilizar una resolución de escaneo de 300ppp.

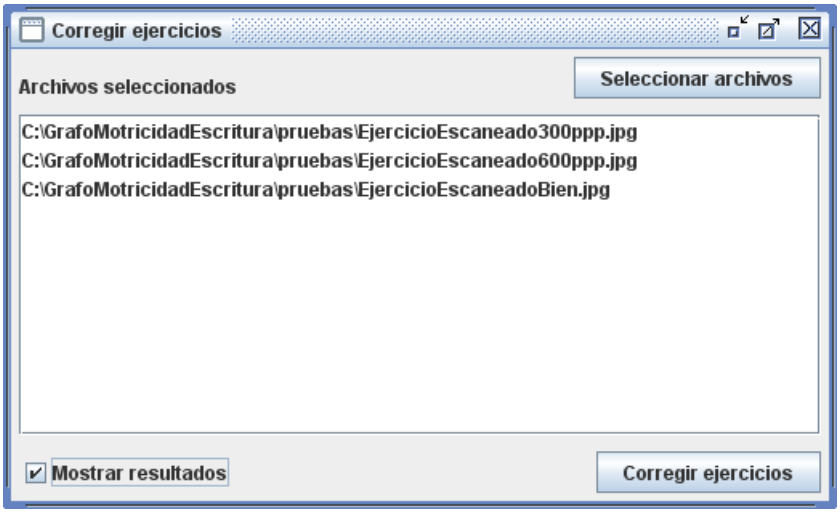

Figura 41: Corrección de ejercicios

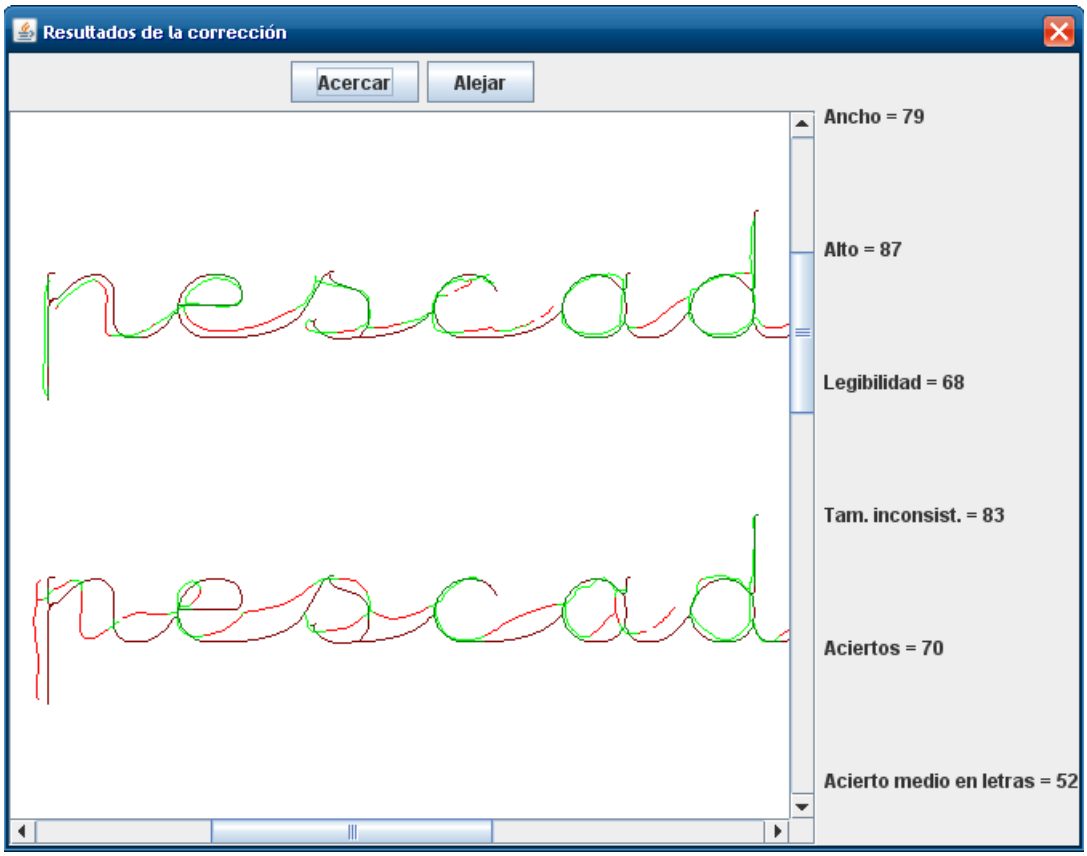

Figura 42: Resultados de la corrección

4. Arquitectura general del sistema

En el capítulo actual se describirán las tecnologías utilizadas para el desarrollo del sistema. Se conocerá el lenguaje Java, que es el que ha sido utilizado, junto con el sistema de base de datos MySQL para la gestión de la base de datos. Además se explicará el diseño del sistema junto con el diseño de la base de datos.

4.1. Introducción al lenguaje de programación Java

Java surgió en 1991 cuando un grupo de ingenieros de Sun Microsystems trataron de diseñar un nuevo lenguaje de programación destinado a electrodomésticos. La reducida potencia de cálculo y memoria de los electrodomésticos llevó a desarrollar un lenguaje sencillo capaz de generar código de tamaño muy reducido.

Debido a la existencia de distintos tipos de CPUs y a los continuos cambios, era importante conseguir una herramienta independiente del tipo de CPU utilizada. Desarrollaron un código "neutro" que no dependía del tipo de electrodoméstico, el cual se ejecutaba sobre una "máquina hipotética o virtual" denominada Java Virtual Machine (JVM). Era la JVM quien interpretaba el código neutro convirtiéndolo a código particular de la CPU utilizada. Esto permitía lo que luego se ha convertido en el principal lema del lenguaje: "Write Once, Run Everywhere". A pesar de los esfuerzos realizados por sus creadores, ninguna empresa de electrodomésticos se interesó por el nuevo lenguaje.

Como lenguaje de programación para computadores, Java se introdujo a finales de 1995. La clave fue la incorporación de un intérprete Java en la versión 2.0 del programa Netscape Navigator, produciendo una verdadera revolución en Internet. Java 1.1 apareció a principios de 1997, mejorando sustancialmente la primera versión del lenguaje. Java 1.2, más tarde rebautizado como Java 2, nació a finales de 1998.

Al programar en Java no se parte de cero. Cualquier aplicación que se desarrolle "cuelga" (o se apoya, según como se quiera ver) en un gran número de clases preexistentes. Algunas de ellas las ha podido hacer el propio usuario, otras pueden ser comerciales, pero siempre hay un número muy importante de clases que forman parte del propio lenguaje (el API o Application Programming Interface de Java). Java incorpora en el propio lenguaje muchos aspectos que en cualquier otro lenguaje son extensiones propiedad de empresas de software o fabricantes de ordenadores (threads, ejecución remota, componentes, seguridad, acceso a bases de datos, etc.).

Por eso muchos expertos opinan que Java es el lenguaje ideal para aprender la informática moderna, porque incorpora todos estos conceptos de un modo estándar, mucho más sencillo y claro que con las citadas extensiones de otros lenguajes. Esto es consecuencia de haber sido diseñado más recientemente y por un único equipo.

El principal objetivo del lenguaje Java es llegar a ser el "nexo universal" que conecte a los usuarios con la información, esté ésta situada en el ordenador local, en un servidor de Web, en una base de datos o en cualquier otro lugar.

Java es un lenguaje muy completo (de hecho se está convirtiendo en un macro-lenguaje: Java 1.0 tenía 12 paquetes; Java 1.1 tenía 23, Java 1.2 tenía 59 y en la última versión (1.6) tiene más de 200 paquetes). En cierta forma casi todo depende de casi todo. Por ello, conviene aprenderlo de modo iterativo: primero una visión muy general, que se va refinando en sucesivas iteraciones. Una forma de hacerlo es empezar con un ejemplo completo en el que ya aparecen algunas de las características más importantes.

La compañía Sun describe el lenguaje Java como "simple, orientado a objetos, distribuido, interpretado, robusto, seguro, de arquitectura neutra, portable, de altas prestaciones, multitarea y dinámico". Además de una serie de halagos por parte de Sun hacia su propia criatura, el hecho es que todo ello describe bastante bien el lenguaje Java, aunque en algunas de esas características el lenguaje sea todavía bastante mejorable.

La máquina virtual de Java

La existencia de distintos tipos de procesadores y ordenadores llevó a los ingenieros de Sun a la conclusión de que era muy importante conseguir un software que no dependiera del tipo de procesador utilizado. Se planteó la necesidad de conseguir un código capaz de ejecutarse en cualquier tipo de máquina. Una vez compilado no debería ser necesaria ninguna modificación por el hecho de cambiar de procesador o de ejecutarlo en otra máquina. La clave consistió en desarrollar un código "neutro" el cual estuviera preparado para ser ejecutado sobre una "máquina hipotética o virtual", denominada Java Virtual Machine (JVM). Es esta JVM quien interpreta este código neutro convirtiéndolo a código particular de la CPU utilizada. Se evita tener que realizar un programa diferente para cada CPU o plataforma. La JVM es el intérprete de Java. Ejecuta los "bytecodes" (ficheros compilados con extensión *.class) creados por el compilador de Java (javac.exe). Tiene numerosas opciones entre las que destaca la posibilidad de utilizar el denominado JIT (Just-In-Time Compiler), que puede mejorar entre 10 y 20 veces la velocidad de ejecución de un programa.

Librerías de Java

En la mayoría de los sistemas operativos actuales, se ofrece una cantidad de código para simplificar la tarea de programación. Este código toma la forma, normalmente, de un conjunto de librerías dinámicas que las aplicaciones pueden llamar cuando lo necesiten. Pero la Plataforma Java está pensada para ser independiente del sistema operativo subyacente, por lo que las aplicaciones no pueden apoyarse en funciones dependientes de cada sistema en concreto. Lo que hace la Plataforma Java, es ofrecer un conjunto de librerías estándar, que contiene mucha de las funciones reutilizables disponibles en los sistemas operativos actuales.

Las librerías de Java tienen tres propósitos dentro de la Plataforma Java. Al igual que otras librerías estándar, ofrecen al programador un conjunto bien definido de funciones para realizar tareas comunes, como manejar listas de elementos u operar de forma sofisticada sobre cadenas de caracteres. Además, las librerías proporcionan una interfaz abstracta para tareas que son altamente dependientes del hardware de la plataforma destino y de su sistema operativo. Tareas tales como manejo de las funciones de red o acceso a ficheros, suelen depender fuertemente de la funcionalidad nativa de la plataforma destino. En el caso concreto anterior, las librerías java.net y java.io implementan el código nativo internamente, y ofrecen una interfaz estándar para que aplicaciones Java puedan ejecutar tales funciones. Finalmente, no todas las plataformas soportan todas las funciones que una aplicación Java espera. En estos casos, las librerías bien pueden emular esas funciones usando lo que esté disponible, o bien ofrecer un mecanismo para comprobar si una funcionalidad concreta está presente.

4.2. Introducción a MySQL

MySQL es un sistema de gestión de base de datos (véase reseña a continuación) relacional, multihilo y multiusuario con más de seis millones de instalaciones. MySQL AB —desde enero de 2008 una subsidiaria de Sun Microsystems— desarrolla MySQL como software libre en un esquema de licenciamiento dual.

SQL (*Lenguaje de Consulta Estructurado*) fue comercializado por primera vez en 1981 por IBM, el cual fue presentado a ANSI y desde ese entonces ha sido considerado como un estándar para las bases de datos relacionales. Desde 1986, el estándar SQL ha aparecido en diferentes versiones como por ejemplo: SQL:92, SQL:99, SQL:2003. MySQL es una idea originaria de la empresa opensource MySQL AB establecida inicialmente en Suecia en 1995 y cuyos fundadores son David Axmark, Allan Larsson, y Michael "Monty" Widenius. El objetivo que persigue esta empresa consiste en que MySQL cumpla el estándar SQL, pero sin sacrificar velocidad, fiabilidad o usabilidad.

Michael Widenius en la década de los 90 trató de usar *mSQL* para conectar las tablas usando rutinas de bajo nivel ISAM, sin embargo, mSQL no era rápido y flexible

para sus necesidades. Esto lo conllevó a crear una API SQL denominada MySQL para bases de datos muy similar a la de *mSQL* pero más portable.

La procedencia del nombre de MySQL no es clara. Desde hace más de 10 años, las herramientas han mantenido el prefijo `My`. También, se cree que tiene relación con el nombre de la hija del cofundador Monty Widenius quien se llama *My*.

Sistema de gestión de base de datos

Los sistemas de gestión de base de datos (SGBD) son un tipo de software muy específico, dedicado a servir de interfaz entre la base de datos, el usuario y las aplicaciones que la utilizan.

Existen distintos objetivos que deben cumplir los SGBD:

- **Abstracción de la información**. Los SGBD ahorran a los usuarios detalles acerca del almacenamiento físico de los datos. Da lo mismo si una base de datos ocupa uno o cientos de archivos, este hecho se hace transparente al usuario. Así, se definen varios *niveles de abstracción*.
- **Independencia**. La independencia de los datos consiste en la capacidad de modificar el esquema (físico o lógico) de una base de datos sin tener que realizar cambios en las aplicaciones que se sirven de ella.
- **Consistencia**. En aquellos casos en los que no se ha logrado eliminar la redundancia, será necesario vigilar que aquella información que aparece repetida se actualice de forma coherente, es decir, que todos los datos repetidos se actualicen de forma simultánea. Por otra parte, la base de datos representa una realidad determinada que tiene determinadas condiciones, por ejemplo que los menores de edad no pueden tener licencia de conducir. El sistema no debería aceptar datos de un conductor menor de edad. En los SGBD existen herramientas que facilitan la programación de este tipo de condiciones.
- **Seguridad**. La información almacenada en una base de datos puede llegar a tener un gran valor. Los SGBD deben garantizar que esta información se encuentra segura frente a usuarios malintencionados, que intenten leer información privilegiada; frente a ataques que deseen manipular o destruir la información; o simplemente ante las torpezas de algún usuario autorizado pero despistado. Normalmente, los SGBD disponen de un complejo sistema de permisos a usuarios y grupos de usuarios, que permiten otorgar diversas categorías de permisos.
- **Integridad**. Se trata de adoptar las medidas necesarias para garantizar la validez de los datos almacenados. Es decir, se trata de proteger los datos ante fallos de hardware, datos introducidos por usuarios descuidados, o cualquier otra circunstancia capaz de corromper la información almacenada. Los SGBD proveen mecanismos para garantizar la recuperación de la base de datos hasta

un estado consistente (ver Consistencia, más arriba) conocido en forma automática.

- **Respaldo**. Los SGBD deben proporcionar una forma eficiente de realizar copias de respaldo de la información almacenada en ellos, y de restaurar a partir de estas copias los datos que se hayan podido perder.
- **Control de la concurrencia**. En la mayoría de entornos (excepto quizás el doméstico), lo más habitual es que sean muchas las personas que acceden a una base de datos, bien para recuperar información, bien para almacenarla. Y es también frecuente que dichos accesos se realicen de forma simultánea. Así pues, un SGBD debe controlar este acceso concurrente a la información, que podría derivar en inconsistencias.
- **Manejo de Transacciones**. Una Transacción es un programa que se ejecuta como una sola operación. Esto quiere decir que el estado luego de una ejecución en la que se produce una falla es el mismo que se obtendría si el programa no se hubiera ejecutado. Los SGBD proveen mecanismos para programar las modificaciones de los datos de una forma mucho más simple que si no se dispusiera de ellos.
- **Tiempo de respuesta**. Lógicamente, es deseable minimizar el tiempo que el SGBD tarda en darnos la información solicitada y en almacenar los cambios realizados.

JDBC

La conectividad de la base de datos de Java (**JDBC ,** Java Database Connectivity) es un marco de programación para los desarrolladores de Java que escriben los programas que tienen acceso a la información guardada en bases de datos, hojas de cálculo, y archivos "planos". JDBC se utiliza comúnmente para conectar un programa del usuario con una base de datos por “detrás de la escena”, sin importar qué software de administración o manejo de base de datos se utilice para controlarlo. De esta manera, JDBC es una plataforma-cruzada.

Una base de datos que conecta con otros programas se llama fuente de datos. Muchas fuentes de datos, incluyendo los productos producidos por Microsoft y Oracle, utilizan ya un estándar llamado conectividad abierta de la base de datos” (**ODBC**, Open Database Connectivity). Mucho de la herencia en la programación en C y de los programas del Perl utilizan ODBC para conectar con las fuentes de datos. El ODBC consolidó mucha de la concordancia entre los sistemas de administración de base de datos. Las estructuras de JDBC están construidas en esta característica e incrementa el nivel de abstracción. Los puentes de JDBC-ODBC se han creado para permitir que los programas de Java conecten con el software compatible ODBC de la base de datos.

JDBC cumple su objetivo mediante un conjunto de interfaces de java, cada una implementada de manera diferente por distintos distribuidores. El conjunto de clases que la componen se denomina el controlador JDBC.

Al construir una aplicación no hay que preocuparse por la creación de las clases que conectaran con la base de datos, la tarea principal de JDBC es ocultar lo específico de cada base de datos y preocuparse sólo por la aplicación.

4.3. Esquema de la arquitectura

En la Figura 43 se muestra un esquema donde se pueden observar las distintas partes del sistema. Por un lado tenemos los dispositivos de entrada, que puede ser la pantalla del tabletPC para realizar ejercicios, un ejercicio escaneado para su corrección (que se le insertará al sistema en forma de imagen digital), o bien teclado y ratón con el que interactuar con la interfaz.

La interfaz es la que se encarga de gestionar los diferentes módulos y conectarlos con el usuario. A partir de ésta se puede acceder a todas las partes del sistema. Además existen módulos que se conectan entre sí sin intervención del usuario, como puede ser el caso de un módulo que acceda a la base de datos.

A continuación se explica brevemente cada uno de ellos:

- **Interfaz**: Es la encargada de conectar al usuario con los módulos, es decir, crea las ventanas necesarias para que el usuario pueda interactuar con ellos, "escucha" los eventos de teclado y ratón y actúa en consecuencia, etc.
- **Plantillas y modelos**: Este módulo administra las diferentes plantillas y modelos que haya diseñado el usuario, los guarda en la base de datos, etc.
- **Asignación de ejercicios**: Permite al usuario asignar ejercicios a los alumnos. Esta asignación se guarda en la Base de Datos y crea los archivos pertinentes.
- **Realización de ejercicios**: Se encarga de que el alumno pueda realizar los ejercicios en el TabletPC o bien corregir los que haya realizado en papel.
- **Metadatos**: Con él se administra todo lo relativo a los metadatos.
- **Workflow**: Permite crear, editar y borrar pasos de workflow y asignarle los umbrales correspondientes.
- **Usuarios**: Administración de los usuarios del sistema y sus permisos.
- **Monitor de resultados**: Permite comprobar el progreso de cada uno de los alumnos, compararlos con otros, etc.
- **Impresión de ejercicios**: Se encarga de imprimir los ejercicios que han asignados a alumnos.
- **Alumnos**: Gestiona todo lo relativo a éstos: datos personales, su nivel, el paso en el que se encuentra cada uno de ellos, etc.

- **Generación automática de ejercicios**: Crea los ejercicios automáticamente para cada alumno teniendo en cuenta las plantillas y modelos disponibles, así como los metadatos correspondientes de éstos y del paso de workflow en el que se encuentra el alumno.
- **Procesado de imágenes**: Procesa los ejercicios escaneados y segmenta lo escrito por el alumno para que pueda ser corregido por el módulo correspondiente.
- **Corrección**: Se encarga de corregir los ejercicios, ya sean imágenes escaneadas o datos provenientes desde la pantalla del TabletPC.

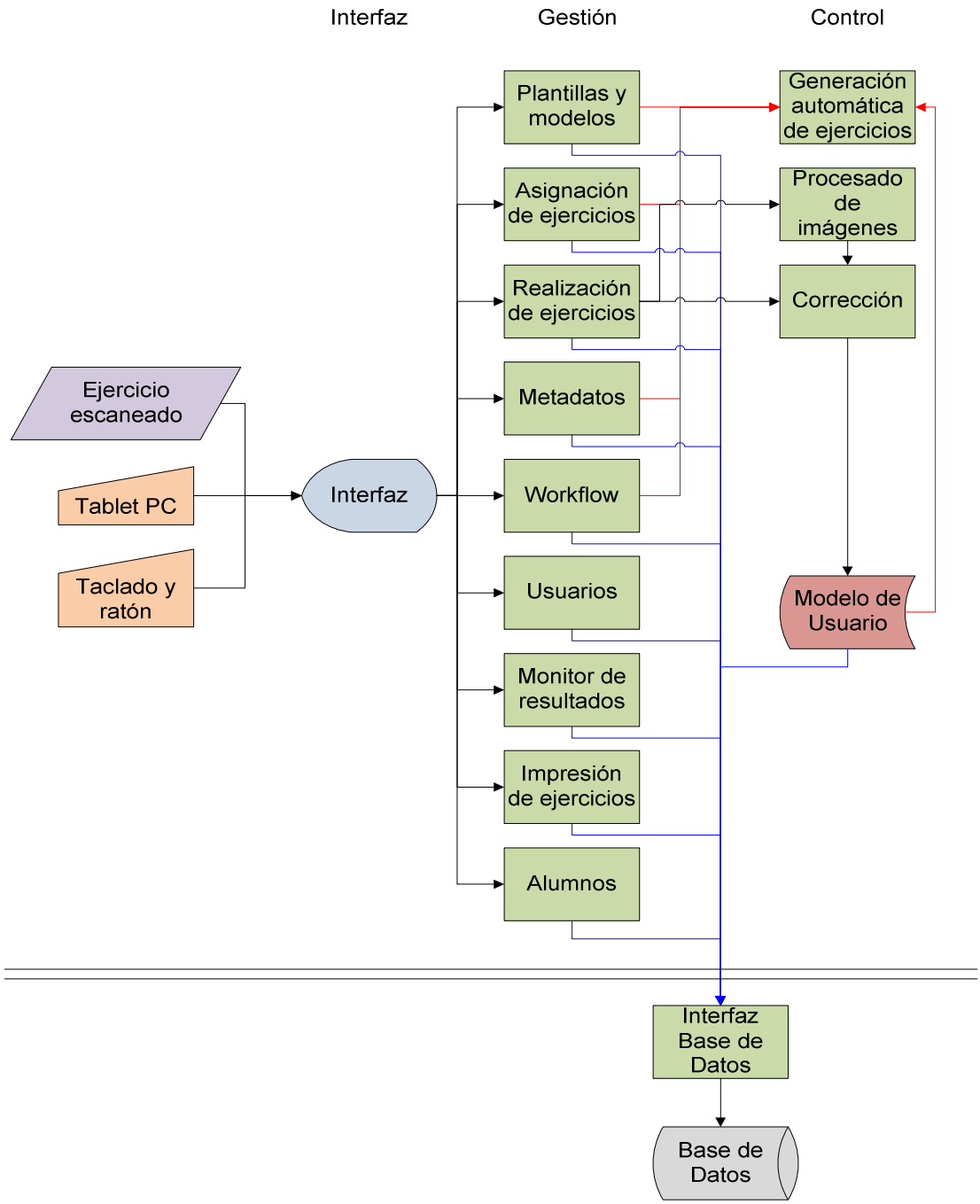

Figura 43: Esquema de la arquitectura del sistema

- **Modelo de usuario**: Gestiona todo lo relacionado con el conocimiento de cada uno de los alumnos.
- **Interfaz de la Base de Datos**: Se encarga de hacer de interlocutor entre el programa y la base de datos.

- **Base de Datos**: Almacena todos los datos necesarios para el correcto funcionamiento del sistema.

4.4. La implementación del sistema y sus clases

En este apartado estudiaremos cómo se han implementado en Java las partes más importantes del sistema, como puede ser la conexión con la Base de datos o la interfaz.

4.4.1. La conexión con la Base de Datos

Debido a que la base de datos se debe consultar desde diferentes partes del sistema por varios motivos (consulta, modificación, inserción, etc.), se ha creado una clase de tipo Factoría. El objetivo es que cualquier clase del sistema pueda acceder a la base de datos fácilmente y con la misma conexión. Es importante para la optimización del sistema que no se vayan creando conexiones innecesarias. En este caso se creará una por cada instancia del programa que esté funcionando. Para ello la clase *DBConnectionFactory* tiene una instancia estática a la clase *DBConnection*, que es la que realmente realiza la conexión. Cuando se le pide a *DBConnectionFactory* la instancia, se comprueba si ya se ha creado. Si es así, simplemente se devuelve. Si no, la crea y a continuación la devuelve. Para que funcione correctamente, antes se le ha debido especificar a *DBConnectionFactory* mediante los métodos apropiados los datos necesarios para realizar la conexión, como puede ser el nombre de usuario, el password, el servidor, el puerto, etc. Esto se realiza al cargar el programa, en la clase principal, por lo que cualquier clase puede obtener la conexión simplemente con llamar al método *DBConnectionFactory.getDBConnection().* Esto devolverá un objeto de tipo DBConnection.

Para realizar cualquier tipo de acción sobre la base de datos, se ha de crear un objeto *Connection* con el driver correspondiente (en este caso el proporcionado por M*ySql*). Adicionalmente por cada actualización o consulta se ha de crear un objeto *Statement*, sobre el cual se realiza la acción. Todos estos pasos necesarios se evitan mediante la clase *DBConnection*, que sirve de "envoltorio" (wrapper). Por tanto para realizar una consulta sólo se necesitan dos líneas: una llamada a *DBConnecionFactory* para obtener el objeto *DBConnection* y otra al método correspondiente de esta clase para realizar la consulta o actualización.

Además se le ha añadido a la clase *DBConnection* los métodos necesarios para modificar la conexión, como puede ser el commit automático (es decir, que por cada acción que realicemos sobre la base de datos, realice el commit justo después), mirar si la conexión está cerrada o abierta, hacer el commit y el rollback. También se le han añadido métodos para recuperar información sobre la conexión, como puede ser el nombre del servidor, el puerto…

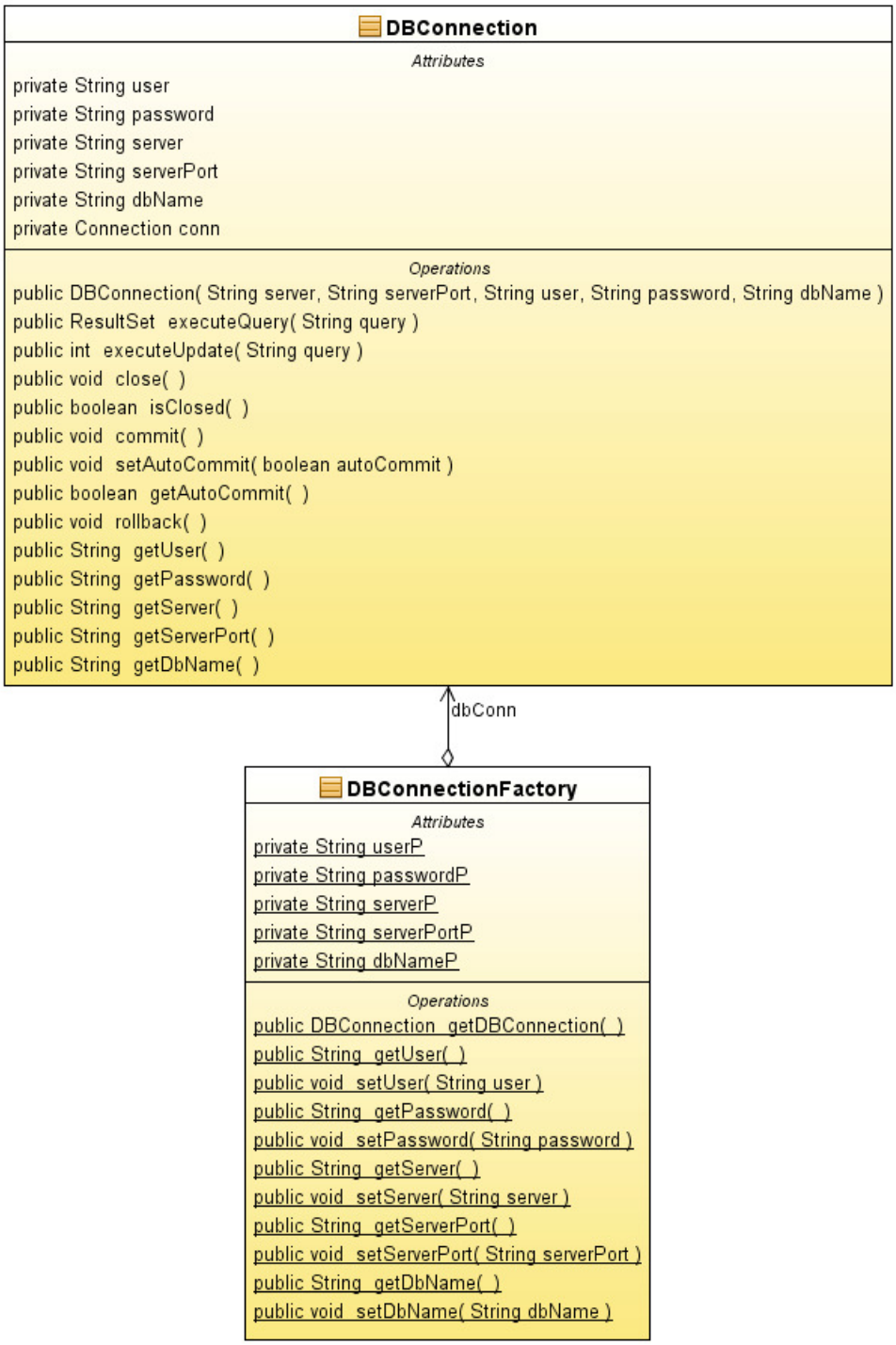

Figura 44: Diagrama de las clases para acceso a la Base de Datos

El esquema se puede ver en la Figura 44. Cabe destacar que todos los métodos de la clase *DBConnectionFactory* son estáticos y las variables son variables de clase, de ahí que estén subrayados en el esquema.

4.4.2. La interfaz de usuario

La interfaz de usuario está basada en una sola clase. Esta clase se encarga de mostrar la ventana principal del sistema, administrar todas las ventanas que pueda abrir el usuario, las cuales se pueden mostrar en ventanas internas, en pestañas o en ventanas independientes. Contiene en la parte superior un menú y una barra de herramientas y en la parte inferior una barra de estado. El diagrama simplificado se puede observar en la Figura 45.

Al cambiar de un sistema de visualizado a otro el contenido de las ventanas se tiene que mantener. Por lo tanto las clases que se muestren en las diferentes ventanas deben ser compatibles con todas ellas, es decir, se debe poder mostrar dentro de una *JInternalFrame*, *JFrame* y en un *JTabbedPane*. Si se lee la documentación de Java se puede observar que a todos ellos se les puede añadir una instancia de la clase *Container*. Para mejorar la funcionalidad se ha decidido implementarlas como una extensión de *JPanel*, que a su vez hereda de *JComponent*, la cual es subclase de *Container*. Además, a estas clases se les debe poder cambiar el idioma sin tener que reiniciar el sistema. Adicionalmente se tienen que poder recargar con los datos necesarios en caso de haberlos cambiado desde otra ventana, por ejemplo, si en una ventana se añade una nueva plantilla y la ventana de generación de modelos está abierta, esa nueva plantilla tiene que estar disponible en la ventana de modelos sin que el usuario deba hacer nada. Para ello se ha diseñado una interfaz que todas las ventanas que se añadan al sistema deben implementar. Es la interfaz *LanguageSelector*. De esta forma, cuando se cambie el idioma en la ventana principal, basta con llamar al método *changeLanguage()* de cada una de los 'containers' abiertos. Un ejemplo de este tipo de clase se puede ver en la Figura 46. Estas clases no sólo se utilizan de interfaz sino que también realizan las operaciones que se esperan de ellas, como generar plantillas, modificar la base de datos, etc. Aunque estas operaciones están convenientemente separadas de las relacionadas con la interfaz. De esta forma, si se desea cambiar determinados elementos para mejorar la interfaz, los métodos de manipulación de datos no se tienen que modificar.

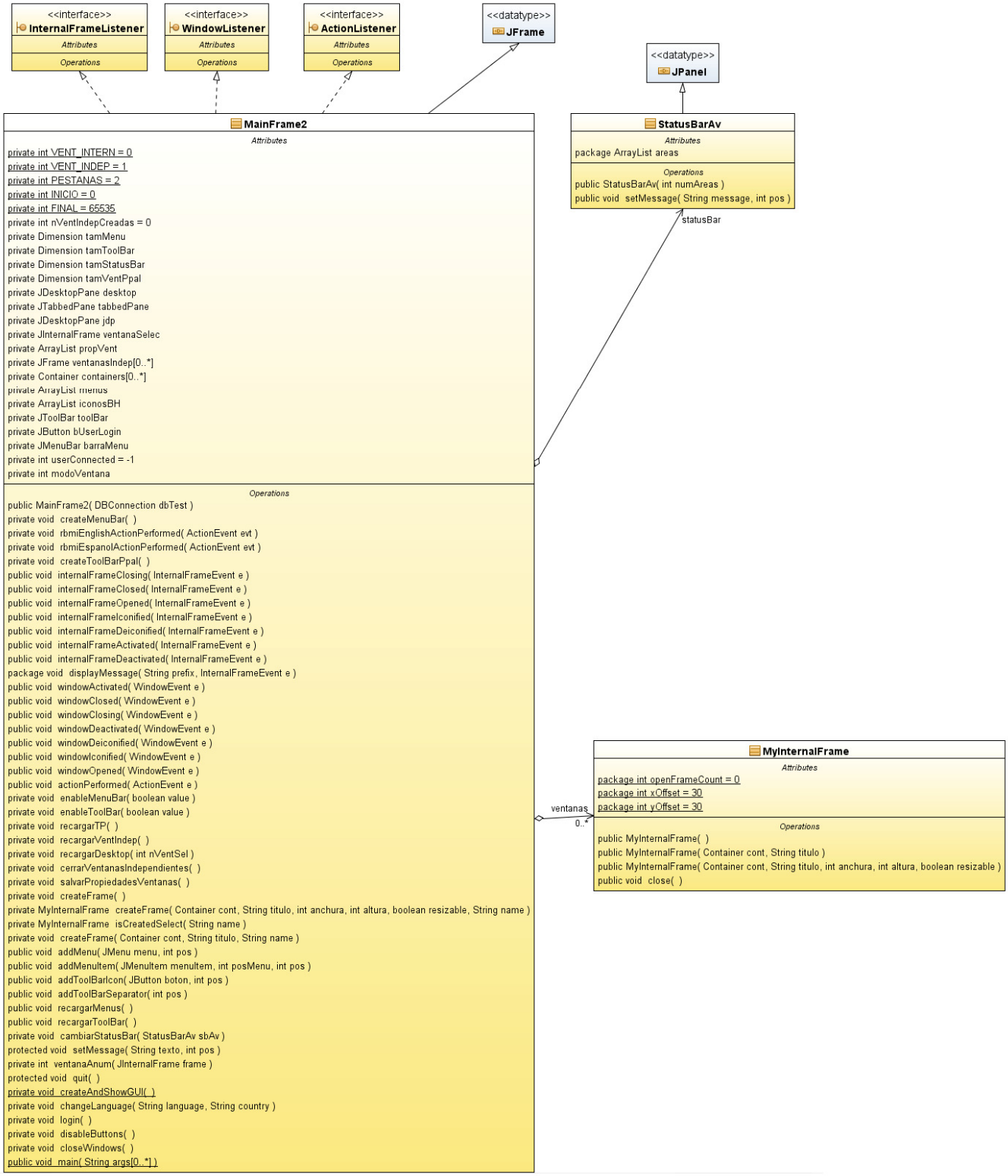

Figura 45: Diagrama de la interfaz principal

Figura 46: Ejemplo de una clase para insertar dentro de las ventanas

4.5. La arquitectura de la Base de Datos

En este apartado se explicarán las tablas de la base de datos y su relación entre sí. Al haberse diseñado de forma iterativa, la base de datos ha ido aumentando su complejidad a lo largo del tiempo, conforme se iba añadiendo funcionalidad. La base

de datos se compone de 20 tablas, las cuales están relacionadas entre sí. Debido a la escasez de espacio va a ser imposible mostrar todo el esquema a la vez, por lo que vamos a dividirlo en partes y explicaremos cada una de ellas por separado.

La normalización de una base de datos consiste en un proceso de análisis de los esquemas de relación dados basado en sus dependencias funcionales y claves primarias para:

- Eliminar la redundancia
- Minimizar las anomalías de inserción, eliminación y actualización

En nuestro caso no se ha llevado a cabo tal proceso debido a dos factores fundamentales: Simplicidad en el esquema de la base de datos (a veces los procesos de normalización pueden introducir cierta complejidad a la hora de interpretarlo) y facilidad de consulta. Además, se ha cuidado su diseño para que este proceso no se tenga que llevar a cabo ya que de otra forma habría que cambiar de nuevo las consultas y actualizaciones del sistema con la consiguiente posibilidad de introducir fallos. Esto, en un desarrollo de prototipos donde la base de datos va cambiando cada cierto tiempo, no es muy recomendable.

4.4.1. La tabla Alumno y sus relaciones

A continuación se describe las tablas que se utilizan para guardar toda la información relativa a los alumnos, desde los datos personales, pasando por los resultados de los ejercicios que han realizado, el paso de workflow en el que se encuentra, etc. El esquema se puede observar en la Figura 47.

Tabla alumno

En la tabla alumno se almacenan todos los datos personales del alumno, como puede ser el nombre, los apellidos, la fecha de nacimiento, la clase en la que se encuentra, etc. Además se guarda el paso de workflow en el que se encuentra y su nivel.

Tabla clase

En esta tabla se guardan las diferentes clases que haya en el centro de educación y el profesor responsable de cada una de ellas.

Tabla users

La tabla *users* se destina a guardar todos los usuarios del sistema junto con su password (encriptado mediante MD5), ya sean alumnos, profesores, administradores, etc. La relación con la tabla de alumnos es obvia, ya que a los usuarios que sean alumnos hay que identificarlos de forma especial para guardar los resultados de sus

ejercicios y evitar que tengan que introducir información adicional para que puedan ser autónomos.

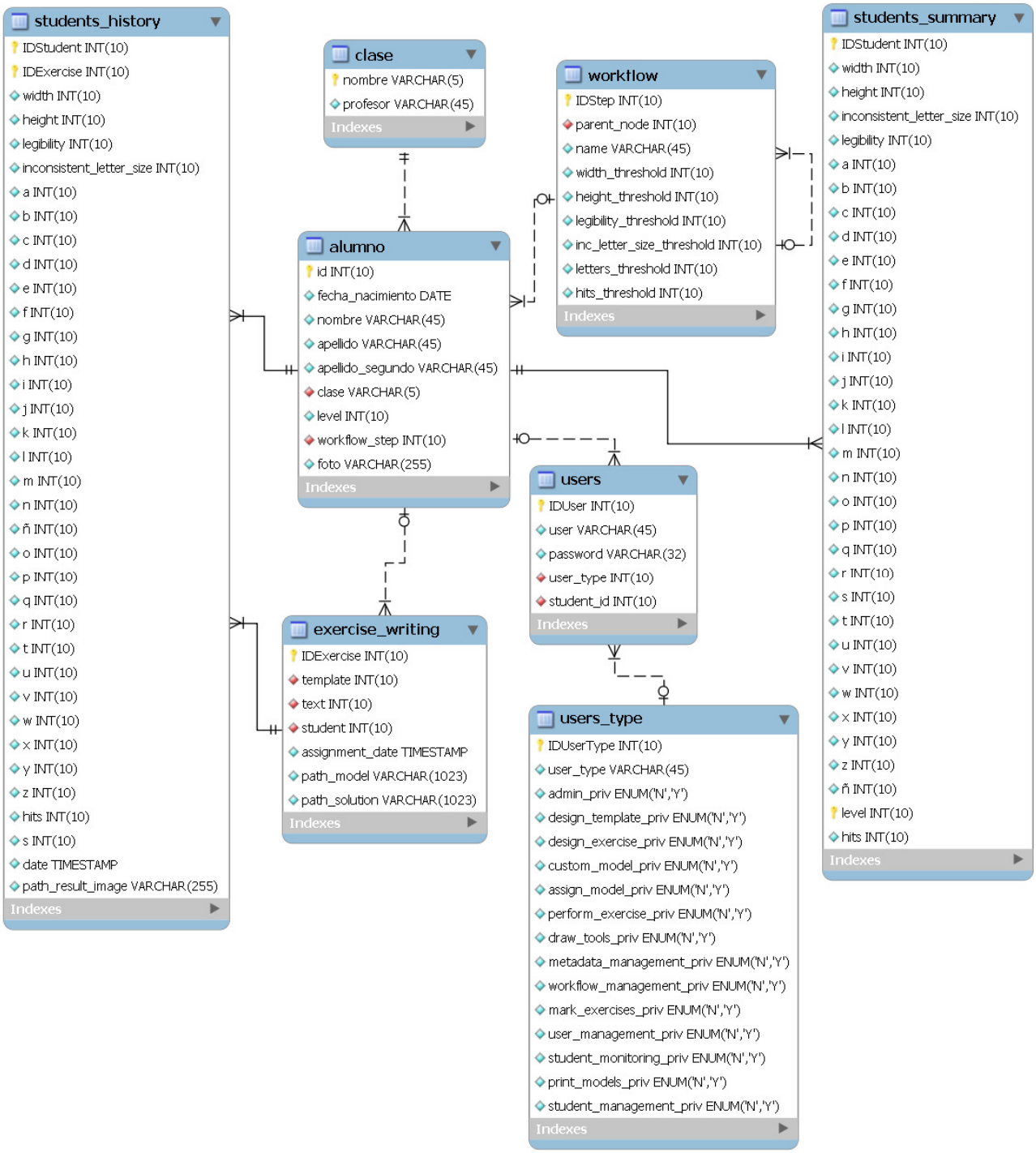

Figura 47: Tabla alumnos y sus relaciones

Tabla users_type

Cada usuario debe tener unos permisos para acceder a determinadas partes del sistema, ya sean específicos o basados en perfiles. En esta tabla se guardan todos los permisos de los usuarios. Si es un perfil, en *user_type* se especificará el nombre del

perfil y en general estará asociado a más de un usuario. En cambio, si se trata de permisos para un usuario individual, no tendrá nombre y sólo estará asociado a un usuario.

Tabla Workflow

En esta tabla se almacenan todos los pasos de workflow que hay en el sistema. Cada uno tendrá su padre, que vendrá especificado en el campo *parent_node*. Además se almacenan los resultados mínimos que debe obtener un alumno para pasar al siguiente paso. Cada alumno podrá encontrarse en uno de estos pasos.

Tablas students_summary y students_history

En estas dos tablas se almacenan los resultados de los ejercicios realizados por el alumno. En *students_summary* se guarda un resumen de los últimos ejercicios realizados, es decir, refleja lo que "sabe" el alumno. La actualización de esta tabla puede hacerse de diferentes formas. Dependiendo de determinados parámetros, el número de ejercicios a tener en cuenta variará, se podrá o no tener en cuenta si se ha pasado de paso de workflow, etc. En realidad la información contenida se ha recuperado de la tabla *students_history*, y podría parecer que es información "duplicada", pero al calcularse de diferente forma en cada momento y no saber a posteriori cómo se ha realizado el cálculo, esta tabla es necesaria.

En *students_history* se guardan todos los ejercicios que ha realizado el alumno, junto con sus valores obtenidos en la corrección.

Tabla exercise_writing

Esta tabla contiene los ejercicios asignados a cada alumno junto con las direcciones de almacenamiento en disco, ya que se almacenan en formato de imagen PNG. De esta forma se pueden imprimir a posteriori y obtener fácilmente la imagen de la solución óptima con la que comparar el ejercicio realizado por el alumno.

4.4.2. Almacenamiento de ejercicios

Como se ha comentado anteriormente, cada ejercicio está compuesto por diferentes elementos. Cada uno de ellos tiene una plantilla y un texto. Además, cada plantilla tiene un formato específico y una configuración de papel. Además, hay que almacenar qué ejercicio se ha asignado a cada alumno, como se ha explicado en la sección 4.4.1, y aquí se muestra la relación con las tablas de ejercicios.

Tabla format

La tabla *format* se utiliza para guardar los diferentes formatos comentados en las secciones 2.3 y 3.3. El formato se define como una línea de texto compuesta por tres tipos de elementos '%d', '%t' y '%l'. Dependiendo del orden de estos, se generará un formato u otro.

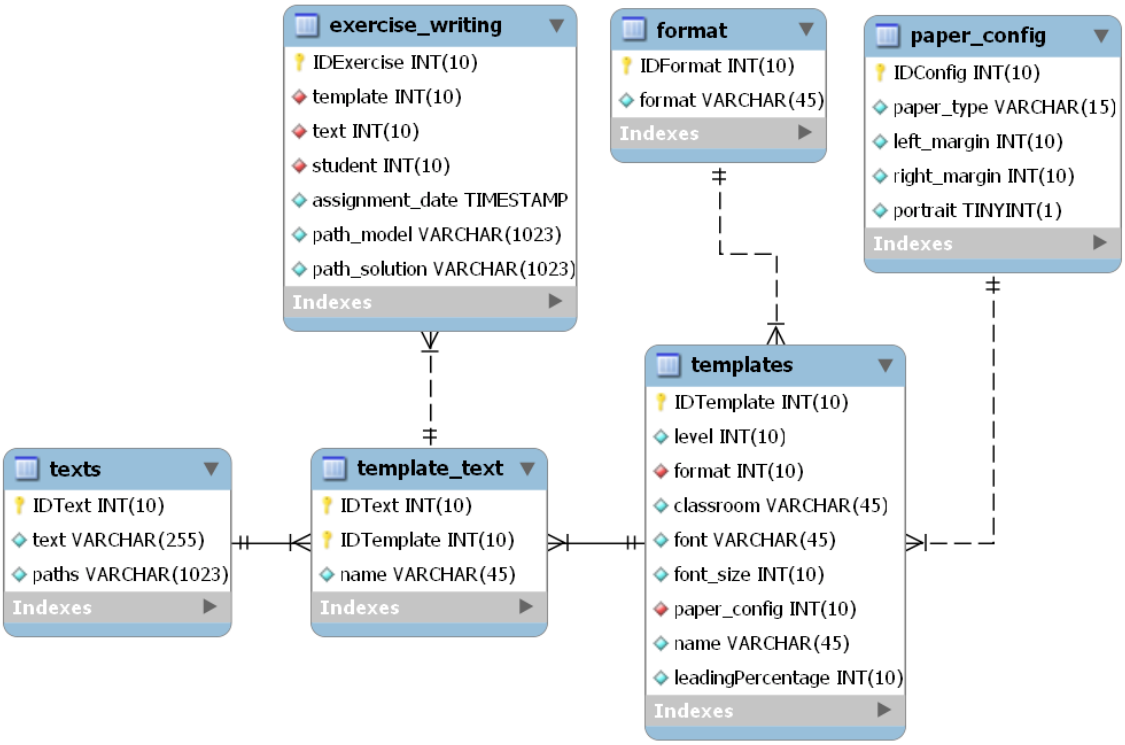

Figura 48: Esquema de la Base de Datos para el almacenamiento de ejercicios

Tabla paper_config

Esta tabla se utiliza para guardar la configuración del papel, es decir, si es tamaño A4 ó A5, los márgenes, etc.

Tabla templates

En la tabla *templates* se guardan las plantillas. Cada una de ellas tiene asociado un formato y una configuración de papel. Además se guardan otras características como el nivel de la plantilla, que lo guardará el profesor dependiendo de la dificultad, la fuente, el tamaño de la misma, la distancia entre líneas, etc.

Tabla text

En esta tabla se guardan los diferentes textos de los modelos. Adicionalmente, también se pueden guardar los paths de las imágenes que van a aparecer en el ejercicio.

Tabla template_text

Esta tabla almacena los modelos, es decir, relaciona una plantilla con un texto y almacena el nombre que le haya dado el profesor.

Tabla exercise_writing

Como hemos comentado en la sección anterior, en esta tabla se almacenan los ejercicios asignados a cada alumno. La relación con la tabla *template_text* es bastante

intuitiva, ya que debe especificar qué ejercicio se ha asignado a un alumno, por lo tanto hace referencia a la plantilla y al texto.

4.4.3. Almacenamiento de los metadatos para el sistema adaptativo

Para que el sistema se pueda adaptar a las necesidades de cada alumno, deben establecerse una serie de características a cada paso de workflow y a cada ejercicio/plantilla para que el sistema pueda reconocer qué tipo de ejercicios están permitidos en ese paso y pueda valorarlos. El esquema se puede observar en la Figura 49. A priori puede parecer que la columna *IDMetadata* de la tabla *metadata_values* es redundante. Su objetivo es poder consultar el metadato al que se refiere el valor, ya que dependiendo del tipo de éste (puede ser un conjunto de valores, un intervalo de enteros, etc.), el valor se almacena en un formato u otro.

Las tablas *templates_metadata*, *workflow_metadata* y *model_metadata* están asociadas a sus respectivas tablas: *templates*, *workflow* y *template_text*.

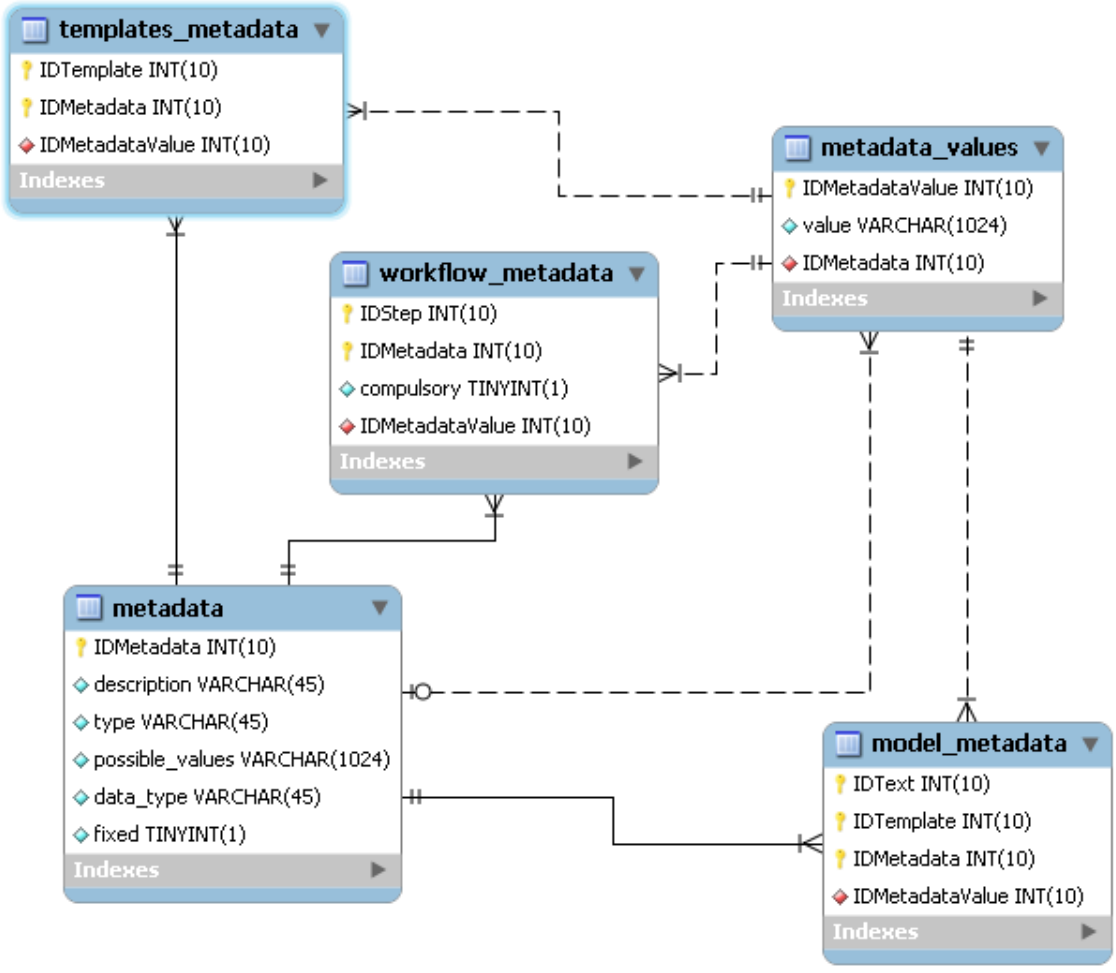

Figura 49: Almacenamiento de metadatos

Tabla metadata

Es la tabla principal y almacena los distintos tipos de metadatos que puede haber, los posibles valores que puede tomar cada uno, su tipo, etc.

Tabla metadata_values

Cada metadato puede tener unos valores determinados, pero al asociar dicho metadato a un elemento (ya sea un paso de workflow, una plantilla o un modelo) se puede seleccionar un subconjunto de los posibles valores. Por ejemplo, si un metadato fuese el tamaño de fuente, que puede tomar valores entre 5 y 50 ([5-50]), al asociarlo a un paso de workflow podemos elegir el intervalo [30-35] para que en ese paso de workflow sólo se puedan asignar ejercicios o plantillas que tengan el tamaño de fuente entre 30 y 35.

Esta tabla almacena esos valores que se le asignan a cada elemento, ya sea un paso de workflow, un modelo o una plantilla.

En caso de que el metadato sea de tipo intervalo de valores, se guarda en formato 'X-Y' donde X e Y son dos números cualesquiera siempre que X <= Y. Si es un conjunto de valores, se almacenan los valores separados por comas, por ejemplo 'azul, rojo, amarillo'. En el caso de que se trate de un metadato de tipo simple, no tiene sentido asociar un valor, ya que lo que le da sentido al metadato es el propio metadato en sí y no los valores asociados.

5. Evolución del sistema

Para desarrollar el sistema se ha optado por un desarrollo basado en prototipos. El propósito de este enfoque es que el usuario final del programa sea una parte activa importante en el diseño del mismo mientras se está desarrollando. Generalmente el cliente final suele especificar unos requisitos que el desarrollador debe implementar. El problema estriba en que a veces el cliente, por diferentes motivos, no es capaz de proveer al diseñador de toda la información deseable, con lo que al final del desarrollo el cliente se da cuenta que faltan elementos, o que la funcionalidad no se adapta exactamente a sus necesidades por lo que hay que rediseñarlo. Esto se evita con el desarrollo basado en prototipos. Al principio del mismo se establece un objetivo general, el cual se va dividiendo en subobjetivos. Estos subobjetivos se van implementando en los prototipos y el cliente se encarga de evaluarlos. Esto permite al cliente ser consciente "en tiempo real" de lo que se está implementando, cómo se está haciendo y si realmente es lo que necesita. Además, generalmente a través de la evaluación pueden ir surgiendo nuevas necesidades, es decir, objetivos que no se habían tenido en cuenta en un principio pero que pueden llegar a ser importantes. También se da el caso de que muchas de las necesidades del cliente deben ser interpretadas por el desarrollador, por lo que éste puede comprobar si su interpretación ha sido correcta en una etapa temprana del desarrollo y poderla corregirla en caso de haber sido errónea a tiempo. Debido a que se está desarrollando una aplicación educativa, este método se complementa con una evaluación formativa.

Existen diferentes tipos de desarrollo basado en prototipos. Nosotros seguiremos un prototipo evolutivo, es decir, al prototipo se le irá añadiendo funcionalidad en cada ciclo y el resultado final será la suma de todas las funcionalidades añadidas. A diferencias de otros tipos, donde se desarrolla muy rápidamente sin tener en cuenta los errores que se puedan producir, el objetivo es que el desarrollo se haga robusto desde un principio e irlo refinando de una forma estructurada. Según (Wilson, Rauch, & Paige, 1992) por cada ciclo se establecen cuatro fases:

- **Planificación**: En esta fase se estudian los requisitos del cliente y cómo se van a enfocar.
- **Implementación**: Se crea el prototipo basándose en los estudios y los análisis de la etapa de planificación.
- **Evaluación**: El prototipo se le entrega al cliente, el cual lo estudia y lo evalúa. En esta fase se comprueba si hay que modificar determinadas

características, si es sistema es demasiado complejo para el usuario y hay que simplificarlo, etc.

- **Aprendizaje**: Se estudia por parte de los desarrolladores los resultados obtenidos en la fase de evaluación, se concreta qué partes del prototipo hay que cambiar, cuáles funcionan correctamente, etc. Estos cambios deben realizarse en el siguiente ciclo junto con los previstos.

En nuestro caso se han realizado tres prototipos, que se han evaluado con alumnos y profesores en el colegio de educación infantil "Los claveles" de Mijas, Málaga. Los alumnos que han participado en la evaluación tenían entre 4 y 5 años.

5.1. Ciclo 0

Este no era un ciclo en sí, es decir, no se iba a crear un prototipo para que lo evaluase el cliente, sino más bien era un estudio de las diferentes librerías de Java que íbamos a necesitar, comprobar su funcionamiento, etc. En la Figura 50 se puede comprobar el aspecto del programa en la conclusión de este ciclo.

5.1.1. Objetivos

- Evaluar las diferentes fuentes con las que realizar los ejercicios
- Imprimir un ejercicio con librerías de Java
- Manejo de Imágenes
- Estudio de aplicaciones de filtros a las imágenes (convolución, detección de bordes, etc.)
- Rotación de imágenes

5.1.2. Conclusión

Al ser un ciclo algo especial las conclusiones no son las mismas que cabe esperar de un ciclo de desarrollo, pero sirvió para contextualizar el problema. Se comprobó que había que corregir algunos errores que tenían las fuentes, tales como que no todas ocupaban el mismo espacio. También fue muy útil como prueba para la aplicación de algoritmos básicos de tratamiento de imágenes.

Desde un punto de vista didáctico, durante este ciclo también se hizo un estudio de campo de cómo trabajaban los niños en el aula.

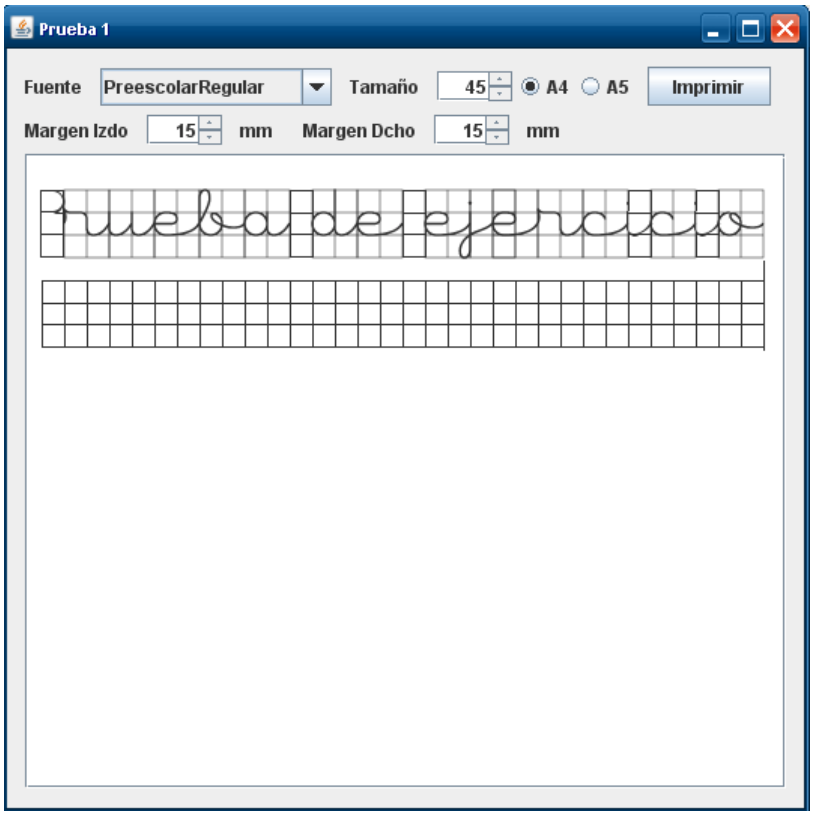

Figura 50: Aspecto del programa a la conclusión del Ciclo 0

5.2. Ciclo 1

Este es el primer ciclo real que va a ser evaluado por los profesores y alumnos. En la Figura 53 se puede comprobar la interfaz del sistema a la finalización del presente ciclo.

A continuación se enumera lo que se pretendía realizar en el presente ciclo:

- Crear una interfaz de usuario sencilla e intuitiva.
- Se debe poder crear un ejercicio manualmente e imprimirlo para que los alumnos puedan realizarlo en papel. Estos ejercicios deben basarse en plantillas y de deben poder asignar a los alumnos.
- Implementar un sistema de internacionalización para que se pueda cambiar el idioma de forma sencilla.
- Impresión en el ejercicio del código de barras que se utilizará posteriormente para la identificación de cada uno de ellos cuando se realizan en papel.

5.2.1. Evaluación de usabilidad y formativa

El prototipo fue evaluado por diferentes profesores para comprobar el funcionamiento. Al principio no se dio explicación algunas a los usuarios para comprobar si el sistema era intuitivo o no. Además se tomó nota de las

recomendaciones para añadir y corregir funcionalidades, tales como que aparezcan dibujos en los modelos, que sean a color, mejora de la interfaz, etc.

También se diseñaron e imprimieron ejercicios con el sistema para que los niños los realizasen sobre el papel. En la Figura 51 se muestran dos de los que se utilizaron para probar algunos algoritmos del sistema.

Figura 51: Ejercicios realizados por alumnos

5.2.2. Conclusiones

Se pudo comprobar cómo la interfaz diseñada no era todo lo intuitiva que se pretendía ni era óptima, ya que si se estaba diseñando un ejercicio y se quería asignar a los alumnos, había que pasar obligatoriamente por la pantalla principal, mostrada en la Figura 53.

Además los ejercicios escaneados no tenían la suficiente resolución para que quedasen bien al imprimirlos. Esto era debido a que la librería de Java para guardar imágenes en formato PNG no guarda en el propio archivo la configuración de la resolución y los ejercicios se veían pixelados. En la Figura 52 se puede comprobar la diferencia.

Figura 52: Comparación de resolución: Antes y después de la modificación

También se apuntó por parte de los docentes la necesidad de ordenar los ejercicios en la pantalla de asignación a los alumnos, porque aunque en el momento

no eran muy numerosos, cuando se creasen los suficientes iba a ser difícil encontrarlos.

Figura 53: Interfaz del sistema a la finalización del ciclo 1

5.3. Ciclo 2

Después del ciclo 1, se corrigieron los errores detectados y se implementaron nuevas funcionalidades, tales como:

- Implementar un modelo de usuario para poder almacenar el conocimiento de cada alumno.
- Proporcionar diferentes métodos de entrada para la creación de los ejercicios, como el tablet PC y la tableta digital.
- Crear un algoritmo de corrección de los ejercicios realizados con los distintos sistemas.
- Crear perfiles de usuario para asignar a cada uno privilegios para acceder a las diferentes partes del sistema
- Crear las pantallas necesarias para la gestión de los estudiantes y alumnos.
- Crear un sistema de fases de aprendizaje en el que cada una de ellas defina unas características que el alumno deberá superar para poder pasar de una fase a otra.

En este ciclo el sistema tenía que quedar casi terminado, al menos debía tener implementado casi todos los requisitos principales para dejar el último ciclo para

refinar más el sistema y no tanto para añadir funcionalidad. Se corrigió la interfaz de usuario y se implementó un sistema de ventanas internas independientes, donde se puede tener cualquier número de ventanas abiertas, las cuales se actualizan automáticamente en caso de que los cambios realizados en una de ellas impliquen cambios en otra (por ejemplo, si se tienen abiertas las ventanas de generar plantillas y modelos y se genera una nueva plantilla, esto se debe reflejar en el combo de plantillas en la ventana de modelos).

5.3.1. Evaluación de usabilidad y formativa

Para la evaluación de este ciclo se crearon determinados ejercicios con la ayuda de los profesores para adaptarlos a los contenidos que en ese momento los alumnos estaban estudiando. Estos ejercicios se realizaron tanto en papel como en el TabletPC. También se probó la tableta digital.

Además, el sistema fue estudiado por docentes para comprobar su funcionamiento, si se habían corregido los problemas presentados en el primer prototipo, etc.

Otro de los objetivos era ver si la organización en plantillas se ajustaba a las necesidades de los niños. Además se estudió si el uso del sistema en el TabletPC era válido para aprender la escritura.

5.3.2. Conclusiones

Se comprobó que la nueva interfaz de usuario (Figura 55) era bastante más fácil de manejar que la antigua aún teniendo mucha más funcionalidad implementada.

Con respecto a los métodos de entrada, el favorito de los alumnos y profesores sigue siendo el papel y lápiz debido a que la pantalla del TabletPC es demasiado gruesa y los alumnos tienen que apretar demasiado para su edad, con lo que el pulso tiembla y por tanto el ejercicio no lo realizan todo lo bien que son capaces. Aunque depende de la fuerza que tuviera el alumno, unos lo realizaban bastante mejor que otros, cuando quizás en papel fuese al contrario. En la Figura 54 se pueden observar a alumnos utilizando el sistema.

Figura 54: Alumnos utilizando el sistema

Se descartó la tableta digital como dispositivo de entrada debido a que los resultados no son representativos ya que es casi imposible mantener el papel fijo a la tableta y por tanto no se almacena exactamente lo que el alumno ha escrito en el papel. Un adulto puede ser consciente del problema y puede bastar con la propia pinza del dispositivo para fijar el papel, pero los alumnos de 4-5 años se apoyan, cambian de postura, mueven la tableta, con lo que resulta imposible. Esto se podía corregir fijando el papel al dispositivo con adhesivos, pero no es nada práctico ya que por cada ejercicio realizado había que aplicar el adhesivo, volverlo a quitar, etc., por lo que se perdía demasiado tiempo con cada uno.

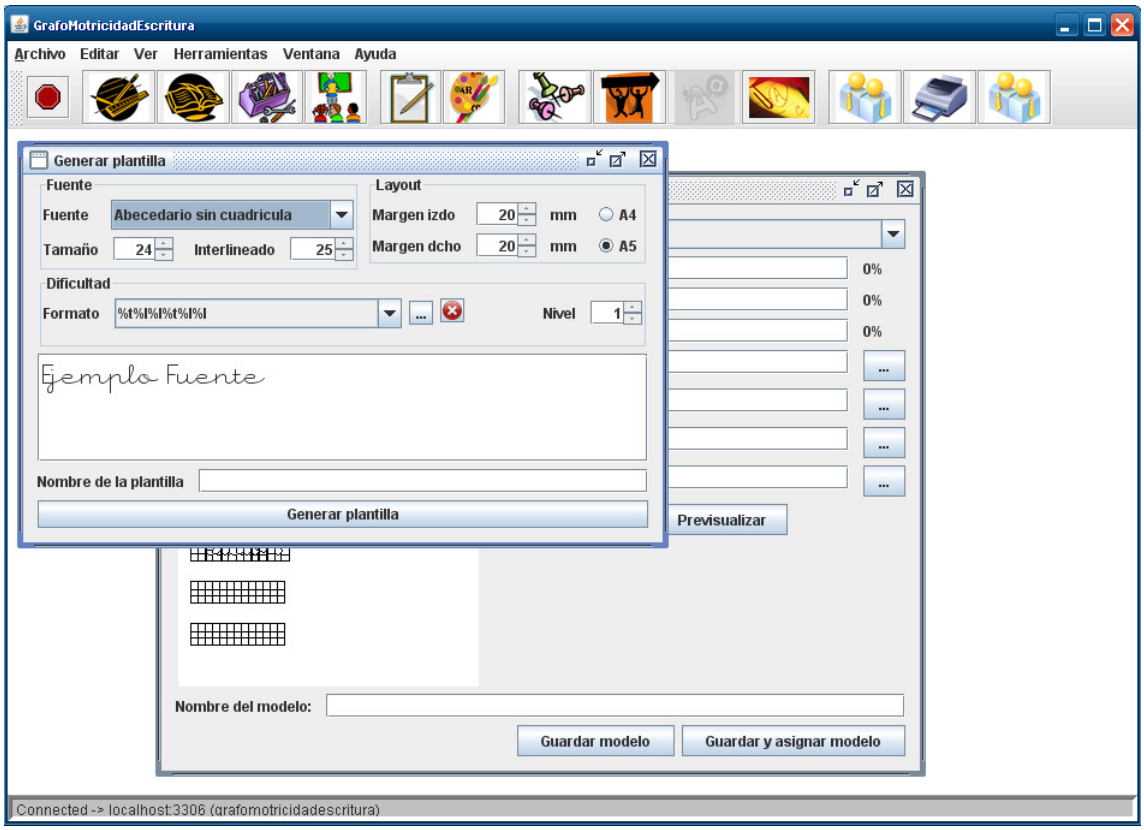

Figura 55: Interfaz de usuario al finalizar el ciclo 2

Se descubrió que el sistema totalmente automático de ejercicios no es viable ya que en determinados momentos el profesor puede querer escoger un ejercicio que no sea el óptimo para ese alumno por diferentes motivos. Así que hay que darle la opción al profesor de elegir entre los distintos ejercicios propuestos por el sistema.

Los perfiles de usuario hay que asociarlos automáticamente a los estudiantes al crear uno nuevo. Además hay que crear un sistema de perfiles para que varios usuarios puedan compartir privilegios y en caso de querer cambiarlos, que sólo haya que cambiarlos del perfil y no a cada uno de los usuarios.

Todas estas correcciones habrá que tenerlas en cuenta en el siguiente ciclo para solucionar todas las incidencias.

5.4. Ciclo 3

En este ciclo se corrigieron todos los errores detectados en los ciclos anteriores y se refinó el sistema. En la Figura 56 se puede observar la interfaz de usuario a la finalización del presente ciclo. Se trata de la misma que la del Ciclo 2, pero con el nuevo botón para monitorizar a los alumnos.

Las funcionalidades a implementar eran:

- La monitorización de los alumnos.
- Dar la opción al educador de elegir entre los ejercicios propuestos por el sistema a la hora de generarlos automáticamente.
- Optimizar el sistema para que consuma menos recursos.
- Implementar modos de generación automática de ejercicios (descrito en el apartado 3.6)

5.4.1. Evaluación

Este prototipo no se ha podido probar con usuarios reales, sino mediante simulación debido a que requieren un ciclo completo de enseñanza.

Se realizaron todo tipo de pruebas para probar el correcto funcionamiento del sistema en versión simulada. Además se comprobó que el sistema va más rápido (por ejemplo, el proceso de crear el ejercicio, guardar la imagen, etc., se redujo a la mitad cambiando y modificando librerías). Se estudió con el 'Profiler' del NetBeans cuáles eran las clases que más espacio ocupaban en memoria y se optimizaron las llamadas.

5.4.2. Conclusión

Se estudiaron los resultados de las pruebas y se vio una clara mejora del sistema. Además se corrigieron determinados fallos que surgieron en las pruebas. Se comprobó que en la monitorización de alumnos faltaba una gráfica que comparase alumnos de distintos niveles, así que se implementó para tener una mejor comprensión de los resultados de los alumnos.

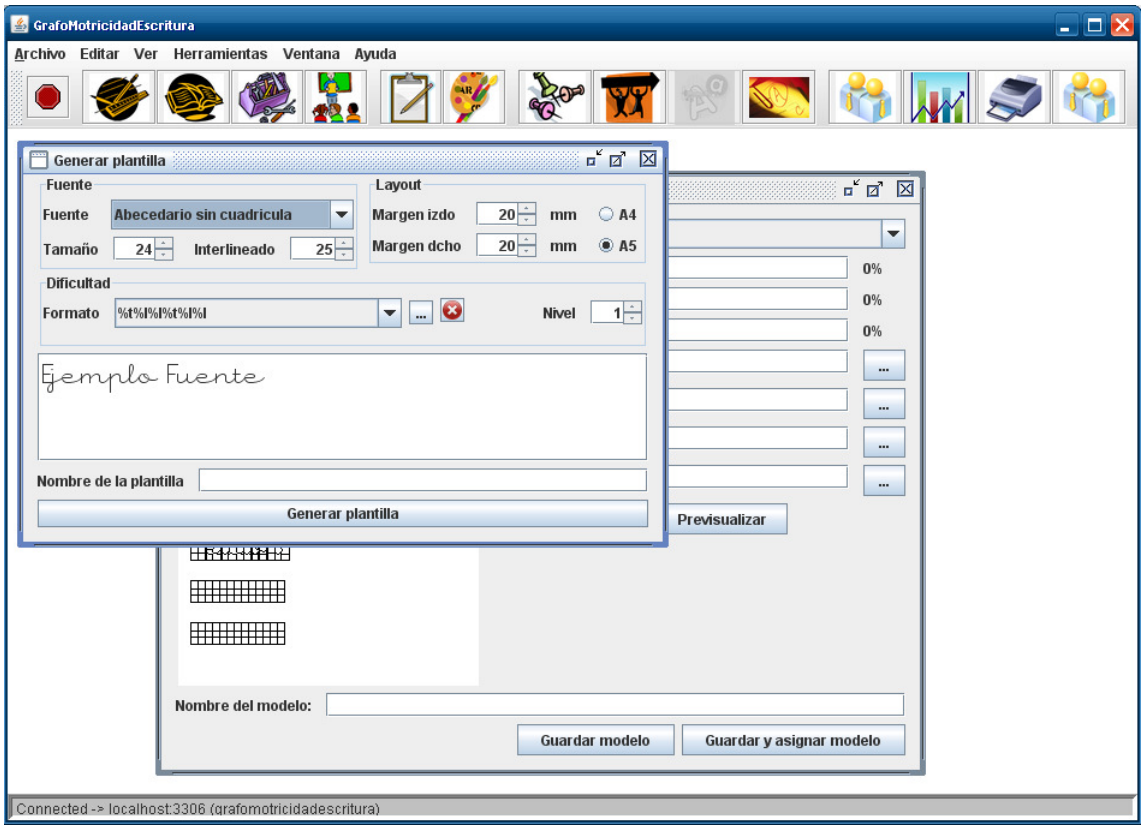

Figura 56: Diseño de la interfaz a la finalización del último ciclo

6. Conclusiones y futuros trabajos

Esta sección pretende servir de resumen al lector para que comprenda en qué consiste el sistema, qué funcionalidades originales aporta (es decir, que no estuvieran ya implementadas en otros sistemas), y en qué se puede mejorar.

A lo largo del desarrollo del trabajo, han ido surgiendo nuevas ideas que por un motivo u otro no han llegado a plasmarse en el resultado final. Hablaremos de ellas, ya que está pensado continuar su desarrollo en un futuro. Muchas de las funcionalidades y del diseño se ha pensado con este fin, para que la mejora y perfeccionamiento del sistema sea lo más cómodo posible.

6.1. Conclusiones, resumen de funcionalidad y reflexiones

En este trabajo **se ha diseñado e implementado un sistema adaptativo de apoyo a la realización y corrección automática de tareas de aprendizaje de la escritura, en educación infantil para la fase de escritura diferenciada**. La creación de este sistema ha supuesto el uso de técnicas de modelado de usuario, modelado instruccional, estudio de sistemas adaptativos así como el uso de diferentes algoritmos de procesado de imágenes digitales, que permiten corregir los ejercicios automáticamente. De este trabajo se ha desarrollado con una metodología centrada en el usuario que ha permitido depurar sus funcionalidades así como diseñar un sistema que realmente se ajusta a las necesidades de los alumnos y profesores de infantil.

El sistema se utiliza como una herramienta para el aprendizaje de la escritura. Para ello se ha estudiado el proceso de aprendizaje en los niños y se han creado los instrumentos necesarios para que los educadores puedan **diseñar todo tipo de ejercicios** y los niños puedan realizarlos. De este modo se pueden personalizar los ejercicios para adaptarlos a cada uno de ellos en vez de utilizar cuadernos fijos para todos.

El sistema incluye funcionalidades para la generación automática de ejercicios de forma **adaptativa** que considera el nivel de conocimiento del modelo de usuario y otros datos del contexto del alumno, para sugerir nuevas tareas. Como soporte a la adaptación:

- Se ha diseñado un Modelo de Usuario para almacenar todos los datos relativos al conocimiento del alumno. Éste se irá actualizando automáticamente al realizar las correcciones de los ejercicios que se le hayan asignado, de forma

manual o automática, independientemente del dispositivo de entrada utilizado.

- Se ha implementado modelo de tareas basado en el concepto de plantilla y modelo con los que se facilita la creación de ejercicios. Dependiendo del diseño de la misma, el ejercicio tendrá una dificultad u otra ya que definirá el tamaño y tipo de letra, el formato del modelo, etc. Con esto se consigue que el educador o el sistema, a partir de una plantilla, pueda generar diferentes modelos con las mismas características básicas, de forma que se puedan adaptar a cada una de las necesidades de los niños sólo con cambiar el contenido. Además, si dos ejercicios tienen la misma plantilla sus resultados se pueden comparar directamente entre sí ya que sus características son similares.

- Se ha desarrollado un modelo de workflow que representa de forma declarativa los pasos (es decir, las fases por las que debe pasar el alumno, en las cuales irá aumentando la dificultad de los ejercicios conforme vaya adquiriendo la destreza necesaria) que marcan el proceso de aprendizaje, según diferentes niveles. Estos pasos vienen definidos por unos metadatos, que se utilizan para fijar determinadas características a cada uno de ellos, como puede ser el número mínimo de ejercicios a realizar, etc.

El tercer elemento representativo del sistema es que los ejercicios son **corregidos automáticamente,** independientemente del dispositivo de entrada utilizado. Se han incorporado algoritmos que permiten evaluar diferentes características para una mejor comprensión de la capacidad del alumno para escribir. Dependiendo del resultado de la corrección, de los ejercicios realizados anteriormente, del paso de workflow en el que se encuentre el alumno, etc., se valora la posibilidad de pasar al siguiente paso. Si la valoración es satisfactoria, el alumno realizará este avance. En caso contrario, deberá hacer más ejercicios hasta que consiga satisfacer los requisitos pertinentes.

Como herramienta de soporte al profesor se ha incorporado un módulo de **monitorización** para el seguimiento del proceso de aprendizaje. Existen diferentes tipos de gráficas para realizar comparaciones, ver el progreso de un alumno de un ejercicio a otro o en el tiempo. Además posee diferentes filtros para poder centrarnos en los datos que deseemos.

El sistema ha sido evaluado con usuarios reales, niños de 4 a 5 años del colegio Los Claveles de Mijas (Málaga). Se ha probado el sistema, estudiado varios dispositivos de entrada, como puede ser el TabletPC, la tableta digital, un ejercicio escaneado, etc. Una vez con los resultados, se decidieron implementar las funcionalidades necesarias

para utilizar aquellos que demostraron ser útiles para una correcta escritura. Al utilizar el TabletPC se obtiene un dato importante, como es el tiempo que tarda un alumno en realizar el ejercicio. Sin embargo, el papel ha demostrado ser el más versátil, no sólo porque todos los alumnos están acostumbrados a él sino porque pueden hacer los ejercicios en paralelo, cosa que sólo se podría hacer con el TabletPC en caso de tener 20-25 unidades, lo que supondría un gasto muy elevado para el colegio. Además, con el papel se pueden llevar los ejercicios a casa en caso de estar enfermos, o por cualquier otra causa.

Finalmente, se ha comparado el sistema implementado con otros existentes para la enseñanza de la escritura. No se ha encontrado ninguno que posea todas las características aquí implementadas. Existen programas que utilizan el concepto de plantilla, como puede ser el abcteach o en sheets, pero no son adaptativos, ni generan ejercicios de forma automática, ni utilizan diferentes dispositivos de entrada. Otros, como el Starwrite o el ComPET están centrados en la escritura de los niños, al igual que el nuestro, pero siguen careciendo de un modelo de usuario en el que apoyarse para realizar una generación automática, si bien es cierto que el ComPET es capaz de evaluar automáticamente las tareas realizadas por los alumnos. Todos ellos tienen la posibilidad de definir nuevos ejercicios para que los alumnos vean cubiertas sus necesidades de aprendizaje y puedan seguir desarrollando sus capacidades, pero de forma manual, es decir sin la componente adaptativa que incorpora nuestro sistema.

6.2.Futuros trabajos

Se pretenden crear más tipos de letras para que los educadores tengan más posibilidades a la hora de realizar los ejercicios. Hay que tener en cuenta que cada letra que se cree, ha de tener una "hermana", la cual será idéntica a la primera pero sin las pautas correspondientes, ya sean cuadrículas, líneas… Esto es esencial para poder crear la solución perfecta con la que comparar el ejercicio realizado por el alumno.

El algoritmo de corrección admite mejoras, sobre todo en lo relativo a la valoración de legibilidad. La idea sería poder valorar cada palabra en este apartado independientemente del tamaño y la posición en la que se encuentre.

La segmentación del ejercicio corregido también se debería mejorar, para que fuese adaptativa. Ahora mismo se han estudiado los parámetros que mejor detectan los tipos de escritura con los diferentes utensilios utilizados hoy en día en los colegios, como pueden ser lápices o bolígrafos. Pero el sistema puede fallar al utilizar rotuladores o ceras de color claro, que no se diferencien demasiado del fondo.

La interfaz puede recibir distintas mejoras para que el usuario tenga más información de lo que está pasando, como aumentar el número de mensajes de la

barra de estado, añadir barras de progreso en los procesos de corrección de ejercicios y en general en aquellos en los que el sistema esté ocupado durante un cierto tiempo.

Al realizar el ejercicio sobre un TabletPC se puede tener en cuenta el tiempo a la hora de valorar un ejercicio. Ya se había pensado en esta solución y ese dato se almacena en cada archivo de log. Habría que hacer un estudio de lo que se considere normal que cada niño debe tardar con un determinado ejercicio, dependiendo del tipo de letra, del tamaño de la misma, etc.

Al ser un sistema que trabaja con imágenes, la manipulación y procesado de éstas puede requerir bastantes recursos del sistema, tanto en memoria como en tiempo de procesador, por lo que se podría estudiar mejor cómo trabaja Java con ellas para optimizar los diferentes procesos.

Índice de Figuras

Referencias

Avgeriou, P., Papasalouros, A., Retalis, S., & Skordalakis, M. (2003). Towards a Pattern Language for Learning Management Systems. *Educational Technology & Society, 6(2)* , 11-24.

de Diego, A. (2009). *Mecanismos para el modelado y monitorización de actividades vinculadas a pre-escritura en educación infantil. Proyecto Fin de Carrera.* Málaga: Universidad de Málaga.

Elmasri, R. A., & Navathe, S. B. (2002). *Fundamentos de sistemas de bases de datos.* Madrid: Addison Wesley.

Feder, K. P., & Majnemer, A. (2007). Handwriting development, competency and intervention. *Developmental Medicine and Child Neurology, 49(4)* , 312-317.

Ferreiro, E., & Teberosky, A. (1979). *Los sistemas de escritura en el desarrollo del niño.* Siglo XXI.

García de Jalón, J., Rodriguez, J. I., Mingo, Í., Imaz, A., Brazález, A., Larzabal, A., y otros. (1999). *Aprenda Java como si estuviera en primero.* San Sebastián: Universidad de Navarra.

Gea, M., & Gutiérrez, F. L. (2002). *El Diseño.* Granada: Universidad de Granada.

González Jiménez, J. (1999). *Visión por computador.* Paraninfo.

Hennion, B., Gentaz, E., Gouagout, P., & Bara, F. (2005). Telemaque, a new visuo-haptic interface for remediation of dysgraphic children. *Proceedings of the First Joint Eurohaptics Conference and Symposium on Haptic Interfaces for Virtual Environment and Teleoperator Systems* (págs. 410-419). IEEE Computer Society.

Henry, K. (Marzo de 2001). *Association For Computer Machinery (ACM)*. Recuperado el 7 de Febrero de 2009, de http://www.acm.org/crossroads/espanol/xrds7-3/ovp_marzo2001.html

Jongmans, M. J., Linthorst-Bakker, E., Westenberg, Y., & Smits-Engelsman, B. C. (2003). Use of a task-oriented self-instruction method to support children un primary school with poor handwriting quality and speed. *Human movement science, 22(4)* , 549-566.

Lingnau, A., Hoppe, H. U., & Mannhaupt, G. (2003). Computer supported collaborative writing in an early learning classroom. *Journal of Computer Assisted Learning, 19(2)* , 186-194.

Millán, E., & Brusilovsky, P. (2007). User Models for Adaptive Hypermedia and Adaptive Educational Systems. En P. Brusilovsky, A. Kobsa, & W. Nejdl, *The adaptive web, LNCS, Vol 4321* (págs. 3-53). Springer Heidelberg.

Morales, C. A. (s.f.). *camoralesm*. Recuperado el 7 de Febrero de 2009, de http://camoralesm.googlepages.com/jdbc

Nikolopoulou, K. (2007). Early Childhood Educational Software: Specific Features and Issues of Localization. *Early Childhood Education Journal, 35(2)* , 173-180.

NISO Press. (2004). *Understanding Metadata.* Bethesda: National Information Standards Organization.

Razton, N. Z., Efraim, D., & Bart, O. (2007). A short-term graphomotor program for improving writing readiness skills of first-grade students. *The american hournal of occuparional therapy, 61(4)* , 399-405.

Read, J. C. (2007). A study of the usability of handwriting recognition for text entry by children. *Interacting with Computers, 19(1)* , 57-69.

Read, J. C., MacFarlane, S., & Casey, C. (2003). A comparison of two online handwriting recognition methods for unconstrained text entry by children. *17th British HCI Conference*, (págs. 29-32). Bristol.

Read, J., & Horton, M. (2004). The Usability of Digital tools in the Primary Writing Classroom. *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications*, (págs. 4386-4391). Chesapeake.

Read, J., Horton, M., & Mazzone, E. (2005). The Design of Digital Tools for the Primary Writing Classroom. *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications*, (págs. 1029-1035). Chesapeake.

Read, J., MacFarlane, S., & Casey, C. (2001). Measuring the Usability of Text Input Methods for Children. *People and Computers XV Joint Proceedings of HCI 2001 and IHM 2001* (págs. 559-572). Preston: Springer-Verlag.

Read, J., MacFarlane, S., & Casey, C. (2002). Pens behaving badly - Usability of pens and graphics tablets for text entry with children. *UIST'02.* Paris.

Read, J., MacFarlane, S., & Horton, M. (2005). The Usability of Handwriting Recognition for Writing in the Primary Classroom. *HCI2004*, (págs. 135-150). Leeds.

Read, J., Mazzone, E., & Horton, M. (2005). Recognition Errors and Recognizing Errors – Children Writing on the Tablet PC. *Lecture Notes in Computer Science, Vol. 3585* , 1096-1099.

Ribera, P. (1999). Leer y escribir: un enfoque comunicativo y constructivista. *Cuadernos de educación, 1* , 13-15.

Rosenblum, S., Parush, S., & Weiss, P. L. (2003). Computerized temporal handwriting characteristics of proficient and non-proficient handwriters. *The American Journal of Occupational Therapy, 57(2)* , 129-138.

Rosenblum, S., Weiss, P. L., & Parush, S. (2003). Product and process evaluation of handwriting dificulties. *Educational psychology review, 15(1)* , 41-81.

Scriven, M. (1991). *Evaluation Thesaurus.* Sage Publications, Inc.

Sim, G., & Horton, M. (2005). Performance and Attitude of Children in Computer Based Versus Paper Based Texting. *Proceedings of World Conference on Educational Multimedia, Hypermedia and Telecommunications* (págs. 3610-3614). P. Kommers & G. Richards.

Smits-Engelsman, B., Niemeijer, A., & van Galen, G. (2001). Fine motor deficiencies in children diagnosed as DCD based on poor grapho-motor ability. *Human Movement Science, 20(1-2)* , 161-182.

Steimle, J., & Brdiczka, O. (2008). Paper-centric interaction concepts for collaborative learning. *Mensch & Computer (MuC) Conference*, (págs. 207-216).

Sun Microsystems, Inc. (2008). *Clase Pattern*. Recuperado el 7 de Febrero de 2009, de Java™ Platform, Standard Edition 6 API Specification: http://java.sun.com/javase/6/docs/api/java/util/regex/Pattern.html

Tolchinsky Landsmann, L. (1993). *El aprendizaje y la enseñanza del lenguaje escrito.* Barcelona: Anthropos.

Wikipedia. (2009). Recuperado el 15 de Enero de 2009, de Plantilla: http://es.wikipedia.org/wiki/Plantilla

Wikipedia. (15 de Enero de 2009). *Wikipedia*. Recuperado el 7 de Febrero de 2009, de http://es.wikipedia.org/wiki/Aplicaci%C3%B3n_Java

Wikipedia. (6 de Febrero de 2009). *Wikipedia*. Recuperado el 8 de Febrero de 2009, de http://es.wikipedia.org/wiki/MySQL

Wilson, D., Rauch, T., & Paige, J. (1992). *Firelili designs*. Recuperado el 13 de Enero de 2009, de http://www.firelily.com/opinions/cycle.html