

## Anexo A

# Planificación del proyecto

---

### A.1. Diagrama de Gantt

La Figura A.1 muestra la organización temporal de las diferentes fases que se han llevado a cabo para el desarrollo de este proyecto.

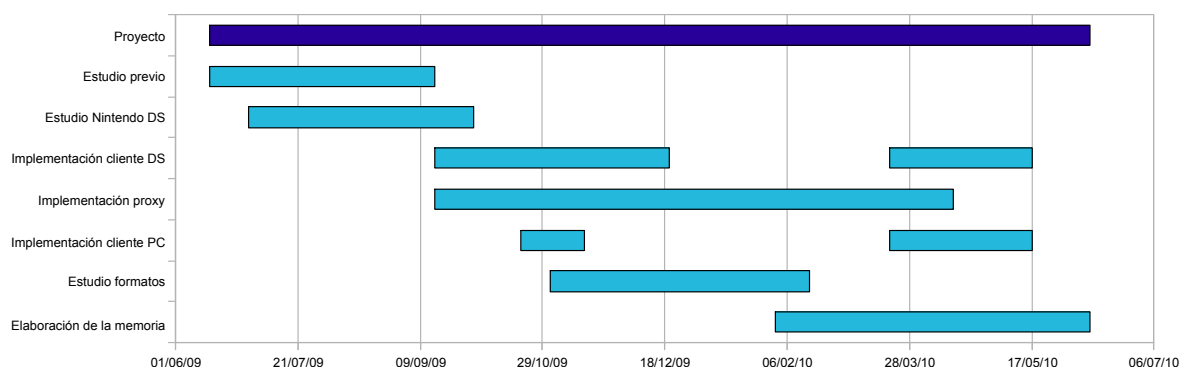


Figura A.1: Diagrama de Gantt

### A.2. Descripción de los pasos seguidos

El proyecto se ha realizado a lo largo de un año, comenzando en junio de 2009 y terminando en junio de 2010. A continuación se describen brevemente los pasos seguidos para su realización.

El proyecto comenzó con un periodo de documentación inicial, con el fin de analizar la viabilidad del mismo. Necesitábamos analizar en profundidad qué queríamos conseguir y cómo queríamos conseguirlo. Se desarrolló la idea de la arquitectura a la vez que se estudiaba la consola Nintendo DS. Era necesario implementar la solución en un cliente y comprobar si la consola podía formar parte de la solución.

Una vez aceptada la consola para su uso en nuestro proyecto, se estudió y se programó para ella con el fin de familiarizarnos con el entorno de desarrollo y el lenguaje utilizado.

Con toda la documentación realizada, el siguiente paso fue la creación de programas de prueba. Se inició al mismo tiempo el diseño del programa cliente para la DS y el proxy, con el fin de implementar la arquitectura y verificar el protocolo de comunicaciones planteado. Una vez asentada la comunicación entre estos dos elementos, se diseñó e implementó el emulador de por-

tarretratos sobre un ordenador portátil.

Con los programas de prueba terminados, se llevó a cabo el estudio de los formatos, que nos llevó algo más de tres meses. Era un punto crítico de la solución y consideramos necesario realizar un análisis detallado.

Con el estudio completado, se incorporaron sus conclusiones a la implementación de los dos programas cliente, con el fin de obtener versiones definitivas y listas para su uso.

Paralelamente, durante los últimos meses, se llevó a cabo la redacción de esta memoria.

## Anexo B

# Viabilidad de formatos

---

En este anexo se recoge el procedimiento llevado a cabo para seleccionar el formato gráfico a utilizar para el envío de las capturas al dispositivo cliente. Consta de un estudio individual de las características principales de cada uno de los formatos seleccionados para, finalmente, realizar una comparativa entre todos los formatos estudiados y poder decidir cual es el formato elegido para la aplicación.

### B.1. Objetivo y metodología

Una parte importante de la aplicación consiste en el envío de imágenes al dispositivo cliente. Toda la información que el cliente va a recibir va a serle proporcionada mediante dichas imágenes. Por lo tanto, este envío se realiza de manera constante y debe ser capaz de proporcionar una buena calidad, pero a la vez latencia corta para no añadir retardo.

El objetivo es seleccionar el formato que, por un lado, codifique la imagen en el menor tiempo posible, de manera que se evite la sobrecarga del servidor y la latencia. Por otro lado, ha de ser un formato con un tamaño reducido para favorecer un envío fluido.

De los distintos formatos existentes para la codificación de imágenes, los más comunes y a la vez más interesantes para la solución son: JPEG, GIF y PNG, ya que ofrecen en principio buenas calidades y tamaños razonables de archivo.

Para tomar la decisión de qué formato es el más apropiado para el proyecto y, por tanto, el que da los mejores tiempos y calidad, es necesario compararlos dentro del entorno de la aplicación. En nuestro caso, se estudian dos entornos, con dos máquinas diferentes. Una de ellas es la Nintendo DS, con una resolución de 256x192 píxeles y la otra un emulador de portarretratos con una resolución de 800x600 píxeles.

Se eligen estos dos entornos ya que se busca un dispositivo móvil con una pantalla aceptable. La Nintendo DS reúne estas características a la vez que posee Wi-Fi para permitir el envío. Su resolución es similar a la de dispositivos cliente alternativos, como puede ser un despertador con pantalla táctil. El segundo entorno se elige pensando en los marcos digitales que están surgiendo en el mercado cuyas pantallas resultan idóneas para este proyecto.

### B.1.1. Objetivo

El objetivo de este análisis es estudiar las características de cada formato, atendiendo a:

- Tiempo invertido en codificar cada imagen en cada formato
- Tamaño de archivo de las imágenes resultantes
- Calidad visual de las imágenes generadas

Estos datos son importantes por diferentes motivos. Por un lado, el tiempo en realizar la conversión es crítico ya que debe ser realizado de manera constante durante la navegación del usuario. Así mismo, el tamaño del archivo es importante, ya que esta imagen es enviada vía red al dispositivo cliente y se requiere hacerlo en el mínimo tiempo posible, con un ancho de banda suficiente. Por último, está el aspecto visual, que si bien es más subjetivo, es importante ya que va a ser lo que el usuario final va a ver. Es de resaltar que el retardo en capturar y transmitir la imagen también acaba redundando en la experiencia del usuario y afecta a la interactividad con la aplicación. Existe un compromiso entre velocidad y calidad global que debe ser tenido en consideración.

Estos factores también influyen en el proxy que, al dar servicio a múltiples dispositivos, puede sufrir sobrecarga.

### B.1.2. Banco de imágenes

Para llevar a cabo un estudio lo más equitativo posible, las capturas que deben ser analizadas han de ser las mismas para cada uno de los formatos. Además, estas capturas deben ser representativas de imágenes que se pueden encontrar en la Web, para dar un resultado objetivo de lo que el cliente va a encontrar en el transcurso de una navegación normal.

Para ello se crea un banco de imágenes de prueba. Como se ha comentado, las pruebas se harán para dos entornos distintos: un dispositivo móvil tipo teléfono, y otro con una pantalla mayor, emulando un marco de fotos. Cada uno de ellos difiere en la resolución y tamaño pantalla, no en potencia de cálculo. Las imágenes seleccionadas serán, por lo tanto, de dos tamaños:

- 256x192 para las pruebas en la Nintendo DS
- 800x600 para las pruebas emulando un portarretratos.

Además, se deciden tomar capturas diferenciándolas en categorías. Esto es así ya que uno de los factores que puede afectar tanto a la calidad del formato como a su tamaño, es su contenido, directamente relacionado con el número de colores que presenta la imagen. Las tres categorías seleccionadas son:

- Capturas con sólo texto y, por tanto, pocos colores
- Capturas con sólo fotografía con muchos colores
- Capturas mixtas foto-texto con una cantidad de colores moderada.

En total se dispone de 30 imágenes, 10 por categoría. Ejemplos de estas imágenes son las contenidas en la Figura B.1



Figura B.1: Categorías de imágenes estudiadas

### B.1.3. Sistema de medición

Para realizar los cálculos de tiempo, se utiliza el *Time Stamp Counter* (TSC) del procesador a través de una unidad para Delphi creada por Udo Jueress. El TSC es un registro de 64 bits que se encuentra en todos los procesadores x86 desde el Intel Pentium. Este registro es el encargado de contar el número de ticks de reloj desde el último reinicio. Esta unidad resulta de gran utilidad ya que es capaz de medir con una precisión de nanosegundos y en algunos casos se necesita medir tiempos de captura y conversión muy pequeños.

### B.1.4. Procedimiento

Para realizar el estudio se decide hacer un programa de prueba enviando imágenes entre el proxy y el cliente con el protocolo de envío establecido, emulando así una comunicación real, pero utilizando el banco de imágenes.

Cada uno de estos formatos dispone de diferentes parámetros para su codificación que afectan a la calidad, al tamaño y al tiempo de procesado. Por ello, es necesario realizar un estudio detallado de cada formato. Este análisis pretende encontrar las características óptimas de cada formato para nuestro juego de pruebas, de forma que podamos decidir cuál se ajusta mejor a nuestras necesidades.

Tomando el banco de imágenes anterior, se comprime cada imagen en el formato correspondiente probando para cada uno de ellos las opciones que tenga disponibles. Una vez realizadas todas las pruebas, se presentan los resultados obtenidos y se decide el formato a utilizar.

## B.2. Descripción y análisis de JPEG

En la siguiente sección se lleva a cabo una descripción y un análisis detallado del formato JPEG y su uso en el proyecto.

### B.2.1. Descripción

Hace referencia a las siglas de Joint Photographic Experts Group, nombre de la comisión que creó la norma. El estándar JPEG especifica tanto la compresión como la descompresión de la imagen. Este estándar es el más utilizado en dispositivos de fotografía.

## Compresión y características

El sistema de compresión que utiliza JPEG es un algoritmo de compresión con pérdida que utiliza 24 bits para codificar cada píxel. Este tipo de compresión hace posible reducir información de la imagen. Es un formato muy interesante para imágenes con grandes degradados o muchos matices de color (por ejemplo gráficos), sin embargo, este formato ofrece malas prestaciones para imágenes con contrastes bruscos (por ejemplo texto).

Este tipo de compresión es viable ya que se basa en las características visuales del ojo humano: notamos mejor los cambios de brillo frente a los cambios de color, y notamos más fácilmente los posibles cambios de brillo en zonas de colores homogéneos. Es importante tener en cuenta que esta compresión es acumulativa, es decir, la compresión sucesiva de una misma imagen provocará pérdidas en cada operación, probablemente degradando su calidad. Este factor es de especial interés ya que la mayoría de las imágenes que se transmitirán provendrán de páginas web con contenido en JPEG, y las vamos a recomprimir.

El grado de compresión puede ser modificado permitiendo así poder variar el tamaño o la calidad, ya que una imagen con compresión mayor será más pequeña, pero esto afectará a su calidad, que será inferior. Gracias a esta opción del formato, es posible encontrar el equilibrio más adecuado para nuestro entorno entre estas dos propiedades.

## Uso en nuestro proyecto

Nuestro estudio pretende hacer una comparativa dentro de este formato entre las diferentes calidades que se pueden emplear, así como efectuar el cálculo de los tiempos derivados de la codificación a este formato y el tamaño de los archivos resultantes.

## Librerías

Las librerías utilizadas son:

- Para codificar JPEG en el proxy y el cliente del portarretratos, programados en Delphi, se usa la librería que viene por defecto para JPEG con Borland Delphi.
- Para decodificar JPEG en el cliente, programado en C++, se usa una librería de descompresión de JPEG creada por Burton Radons para GBA y modificada por Headspin para funcionar en la NDS. La ruta de descarga es <http://www.headsoft.com.au/download/nds/jpeg-decode-ds.zip>.

### B.2.2. Análisis

Siempre vamos a encontrar algún tipo de artificio tras una compresión con pérdidas. Este efecto es más acusado en conversiones de calidad baja, que corresponde con las imágenes de menor tamaño. De forma que existe un compromiso a tener en cuenta entre tamaño y calidad.

Para analizar ese compromiso, se toma el banco de imágenes y se realizan las conversiones para los dos formatos de estudio, tanto Nintendo DS como un formato a 800x600, tomando para ello diferentes grados de calidad, de 10 % a 100 % en intervalos de 10 % para observar el resultado final en calidad, tiempo y tamaño.

Para dar un ejemplo del tipo de pérdidas que vamos a encontrar con este formato, se presenta en la Figura B.2, una imagen con formato JPEG a una calidad del 50 %. En ella se han resaltado dos detalles con una ampliación en relación 1:3.

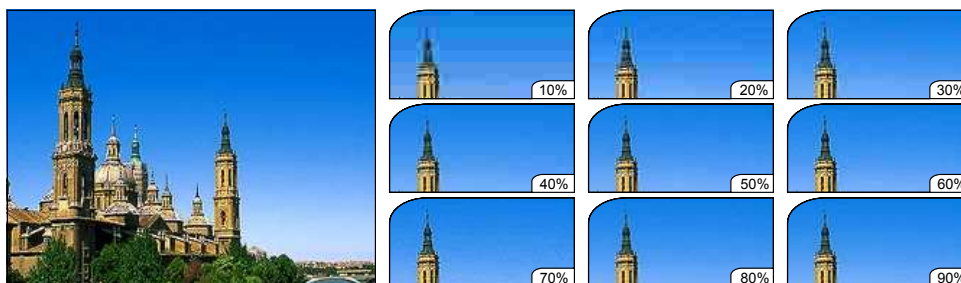


**Figura B.2:** Artificios generados por JPEG

A un 50 % de calidad, se observan irregularidades tanto en la parte de imagen como en la de texto. En la imagen vemos como aparecen píxeles fuera de sitio que producen una visión borrosa. En el texto, además del problema anterior, vemos como los colores de las letras que debían ser homogéneos, presentan diferencias de tonalidad.

### Calidad visual

Antes de estudiar los datos numéricos, es importante estudiar las imágenes generadas ya que es el resultado visual lo que el usuario va a percibir. Buscamos una calidad aceptable y la más cercana al original posible. Las Figuras B.3, B.4 y B.5 muestran una serie de capturas con diferentes calidades poniendo como ejemplo una de las 10 imágenes *tipo foto* analizadas. Por tamaño de la memoria se muestran las imágenes a 256x192, pero las capturas a 800x600 son análogas.



**Figura B.3:** Evolución de la calidad en captura *tipo foto*

La Figura B.3 es un ejemplo de captura con sólo imagen fotográfica. A la izquierda está el JPEG generado con una calidad del 100 % y a la derecha el detalle de las diferentes calidades estudiadas. Vemos cómo, a partir de una calidad del 50 % el resultado comienza a ser satisfactorio

para este tipo de imágenes. Los píxeles fuera de lugar no se perciben.



Figura B.4: Evolución de la calidad en captura mixta

Para las capturas con combinación de texto y foto, como la Figura B.4, la calidad óptima se encuentra en torno al 60-70 %. Ya que como se ha comentado, las imágenes generadas con menos calidad hacen aparecer el texto borroso.



Figura B.5: Evolución de la calidad en captura *tipo texto*

Y por ultimo, para las imágenes con sólo texto, poniendo como ejemplo la Figura B.5, la calidad es aceptable cuando llegamos a un 60-70 %, en ese momento los posibles artificios que pueda haber resultan prácticamente imperceptibles.

**Conclusión** Comparando los tres tipos de imágenes, podemos concretar que, a partir de un porcentaje del 70 % la calidad es perfectamente aceptable y no merece la pena sacrificar el tamaño para buscar una calidad mejor.

## Resultados numéricos

Las gráficas siguientes, muestran el resumen de los resultados obtenidos en media para cada uno de los conjuntos de imágenes de cada tipo que hemos explicado anteriormente para los dos factores de estudio:

**Tiempo de compresión** La Figura B.6 muestra la variación del tiempo de compresión respecto a la calidad. En el eje de las abscisas están representadas las diferentes calidades del JPEG en tanto por ciento y en el eje de ordenadas el tiempo, en milisegundos, que se tarda en codificar la imagen. Estos datos son siempre tomados de la media del conjunto de imágenes seleccionadas para cada tipo, mostrado en la gráfica con una barra para cada una de estas medias.



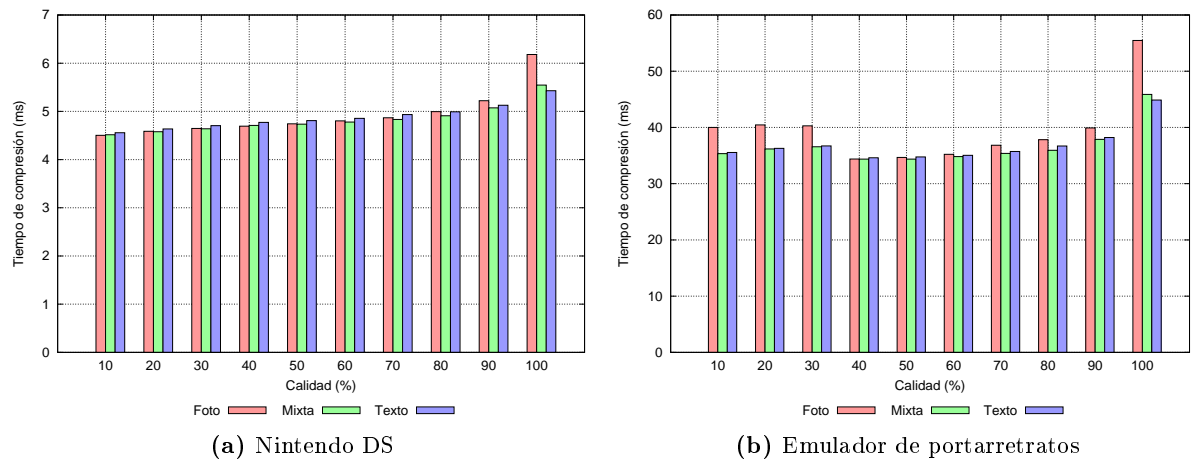


Figura B.6: Tiempo de compresión para formato JPEG

Se observa cómo la variación de este tiempo es mínima y se aproxima a los 5 milisegundos al aumentar la calidad, sobrepasando este valor para las calidades de 90 y 100 %. Además, vemos que el tiempo de compresión es independiente del tipo de imagen, ya que los resultados son similares para cada una de las tres categorías estudiadas. Así que el tiempo no va a ser un dato relevante en la decisión final. Lógicamente, el tiempo de compresión sí depende del tamaño en píxeles de la imagen.

**Tamaño** La Figura B.7 muestra cómo afecta la calidad en el tamaño final de la imagen. En el eje de las abscisas están representadas las diferentes calidades del JPEG en tanto por ciento y en el eje de ordenadas el tamaño en bytes de las imágenes resultantes, en media, de cada conjunto de cada tipo. Las barras muestran cada una de estas tres medias.

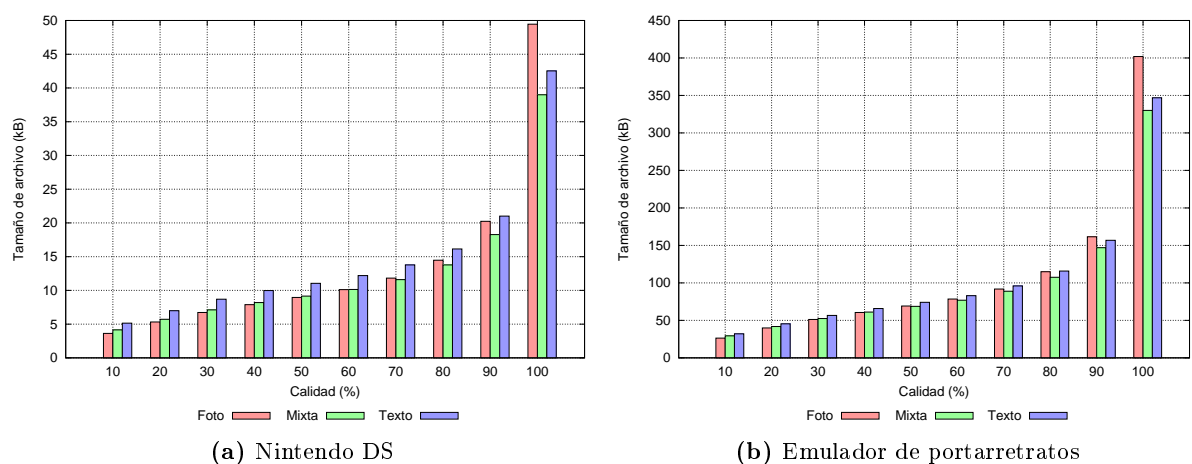


Figura B.7: Tamaño de archivo para formato JPEG

Se observa cómo, cuando la calidad es máxima (100 %), el tamaño de la imagen generada es muy elevado, creando imágenes de 43,6 KB en media. Sin embargo, este salto sólo se produce con el mayor valor de calidad. En el resto de valores, el aumento del tamaño, si bien es mayor

cuanta mayor calidad, aumenta de manera más gradual.

**Conclusión** Teniendo en cuenta los análisis anteriores, se concluye que tanto por tiempo y tamaño como por calidad visual, la calidad elegida para JPEG es del 70 %.

### B.3. Descripción y análisis de GIF

En la siguiente sección se lleva a cabo una descripción y un análisis detallado del formato GIF y su uso en el proyecto.

#### B.3.1. Descripción

Graphics Interchange Format (GIF), es el nombre de un formato introducido por CompuServe en 1987 para la compresión de imágenes, que sigue usándose en la actualidad.

##### Compresión y características

GIF utiliza el algoritmo de compresión sin pérdida LZW (Lempel-Ziv-Welch). El formato GIF hace uso de una paleta para almacenar los colores de cada imagen de entre 2 y 256 colores. Así pues, utiliza color indexado, es decir, la información almacenada en cada píxel es un índice, codificado con 8 bits, a una posición de la paleta en la que se encuentra el color de dicho píxel.

Este tipo de compresión resulta muy conveniente cuando se comprimen imágenes con pocos colores, con grandes secciones de color homogéneo o texto sobre fondo simple, ya que permite reducir el tamaño sin degradar la calidad visual. Sin embargo, la limitación en los colores que puede contener la paleta, hace que no esté recomendado para fotografías o imágenes con mucha variación de color. Para este tipo de imágenes es más adecuado usar formatos como JPEG o PNG.

Para este proyecto, este formato resulta muy interesante ya que dispone de gran cantidad de opciones configurables para ajustarlo a nuestras necesidades. Además es un formato que en principio tiene un tamaño reducido, lo cual resulta interesante para reducir el tiempo de envío.

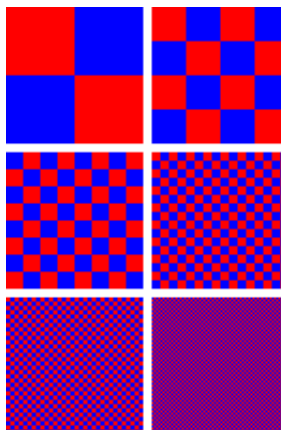
Una de las características que lo distinguen frente al JPEG es la posibilidad de permitir animaciones y transparencias. Las animaciones constan de un conjunto de imágenes, cada una de ellas con su propia paleta de 256 colores.

Para seleccionar el tipo de formato a elegir hay que tener en cuenta, además de las paletas, el *dithering*. Para los colores no representables con la paleta se debe realizar un proceso de traducción o *dithering*.

##### *dithering*

El término *dithering* da nombre al método que consiste en la aproximación de un color no disponible en la paleta, o en un sistema, aproximando píxeles de colores que sí se encuentran en ella.

Cuando un sistema se encuentra con un color que no puede reproducir o cuando se comprime una imagen en un formato como es el GIF con una paleta reducida, se necesita realizar un proceso de *dithering*. Este proceso consiste en intentar simular el color no disponible mezclando píxeles de diferentes colores, generando un color similar al original pero con la inevitable pérdida de calidad visual. Para ilustrar este método, la Figura B.8 muestra cómo, para obtener un color violeta, se recurre a la mezcla de píxeles rojos y azules:



**Figura B.8:** Ejemplo de *dithering*. [http://en.wikipedia.org/wiki/File:Dithering\\_example\\_red\\_blue.png](http://en.wikipedia.org/wiki/File:Dithering_example_red_blue.png)

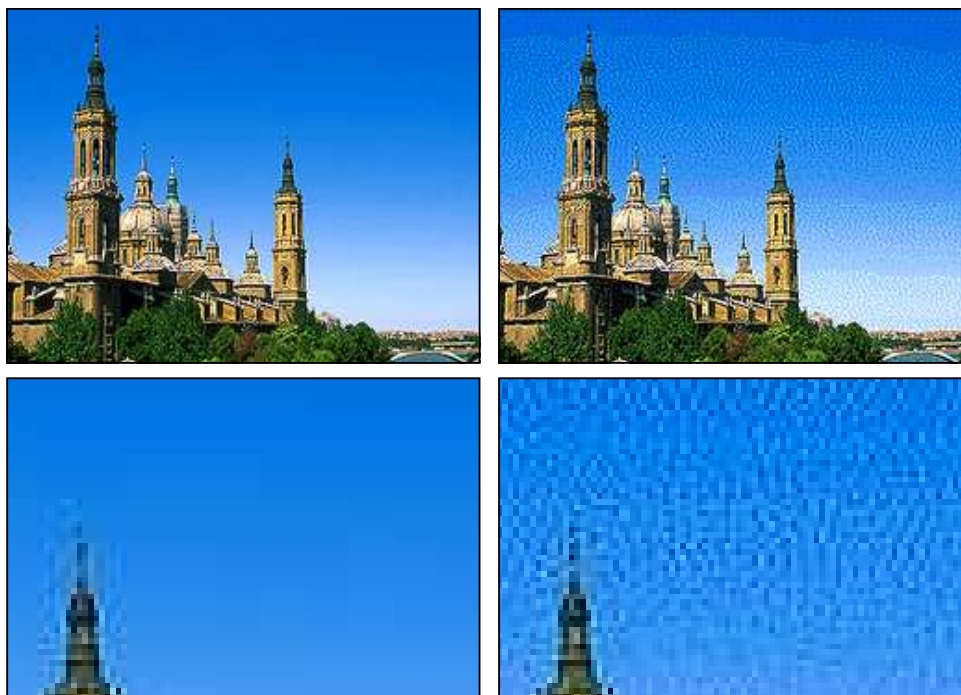
Como ya hemos visto, tenemos dos casos de *dithering*. En primer lugar está el *dithering* necesario para poder transformar una imagen con una cantidad de colores cualquiera en una imagen de únicamente 256 colores. Para ello, el formato GIF hace uso de diferentes algoritmos. De entre todos ellos, estudiaremos los siguientes:

1. Nearest. Realiza un ajuste al color más cercano de la paleta, sin corrección de error.
2. Floyd Steinberg. Sólo difunde el error a píxeles vecinos. Esto genera un *dithering* de grano muy fino.
3. Stucki. Similar al anterior pero más rápido. Suele generar una imagen limpia.
4. Sierra. Basado en Jarvis pero más rápido.
5. Jarvis, Judice, and Ninke difunde el error también al siguiente nivel de píxeles. Es más lento que el Floyd–Steinberg porque distribuye errores a 12 píxeles alrededor y no a 4 como éste.
6. Stevenson-Arce. Algoritmo lento pero, de buena calidad.
7. Burkes. Es una forma simplificada de Stucki que es más rápida pero con resultados menos limpios.

Otro caso de uso de *dithering* se produce cuando algún color de una imagen no puede ser representado por un sistema operativo o un navegador debido a su profundidad de color. Por ejemplo, una imagen en millones de colores reproduciéndose en un sistema con una profundidad de 8 bits (256 colores). En ese caso el *dithering* afectaría a la reproducción de la imagen y no a su codificación, como ocurre por ejemplo al cargar un fondo con más de 256 colores en la Nintendo DS, que utiliza una paleta de 256 colores.

Aun así, un color que ha sufrido el proceso de *dithering* puede pasar inadvertido en una fotografía. En superficies grandes de tonalidades homogéneas, puede alterar en gran medida el color. Por ejemplo un fondo liso. Para ayudarnos a evitar este segundo tipo de *dithering*, existen las paletas ya definidas.

La Figura B.9 pretende explicar el resultado del *dithering* sobre una imagen real, de las que usa la aplicación. Se observa en la parte izquierda la imagen original y a la derecha la imagen después de ser sometida a un proceso de *dithering*. Se aprecia como el cielo en lugar de tener un color degradado homogéneo, presenta píxeles intercalados.



**Figura B.9:** Detalle de *dithering*

## Paletas

La elección de paleta resulta importante, ya que de ella va a depender no solamente el tiempo empleado en codificar la imagen sino la calidad que vamos a obtener.

Como ya hemos visto, el formato GIF sólo puede usar 256 colores, los cuales se encuentran en su paleta codificada en la imagen. Los colores que componen la paleta pueden provenir de alguna de las paletas por defecto o pueden generarse eligiendo una paleta óptima con los colores de la propia imagen.

El uso de las paletas por defecto resulta interesante cuando se trabaja con sistemas con una profundidad de color limitada, como son pantallas de 256 colores. Entre estas paletas destaca la denominada “*web safe*” o “paleta Web” que consta de 216 colores que pueden ser reproducidos por cualquier sistema sea cual sea su profundidad de color, y por todos los navegadores.

Hoy en día, esto no tiene demasiada importancia ya que la mayoría de los equipos son capaces de reproducir imágenes con millones de colores, pero puede resultar útil el uso de una

paleta fija, por ejemplo, si se permite enviar sólo una parte de la imagen. Se ha implementado una optimización por la que se puede solicitar al proxy sólo el área desconocida al realizar un pequeño desplazamiento de la imagen. Si la paleta no es fija para todas las imágenes, sino elegida de forma independiente para cada una de ellas, esta optimización no es posible, ya que los índices de los colores no se corresponderán.

Vamos a estudiar tres paletas, que son las más indicadas para nuestro proyecto:

- **Paleta Windows 256**

Se trata de una paleta con los 256 colores fijos que pueden ser reproducidos por cualquier sistema con Windows, incluso con una profundidad de 8 bits (256 colores).

- **Paleta Web**

Consta de 216 colores. Como ya hemos visto, los colores contenidos en esta paleta pueden ser interpretados por cualquier sistema operativo o navegador de la misma manera.

- **Paleta optimizada**

Se obtiene seleccionando 256 colores contenidos en la imagen. La ventaja evidente de esta paleta es que la calidad será mayor, ya que se aprovechan los 256 colores para aquellos que realmente están en la imagen, o minimizando el error si se utiliza *dithering*.

Puede verse un ejemplo de cómo una imagen, al ser codificada con cada una de estas paletas da como resultado imágenes con calidades distintas en la Figura B.10.



**Figura B.10:** Codificación de imágenes dependiente de la paleta

## Uso en nuestro proyecto

Nuestro estudio consiste en una comparativa de los diferentes tiempos obtenidos en codificar el formato GIF con cada una de las diferentes paletas y el tamaño de los archivos resultantes, así como observar la calidad de la imagen resultante. Para ello usaremos alguna de las paletas predefinidas más usadas, combinándolas con los diferentes métodos de *dithering* antes mencionados.

## Librerías

Las librerías utilizadas para codificar y decodificar GIF son:

- GIFImage creado por Anders Melander, ya integrado en Delphi para codificar GIF en el proxy y decodificarlo en el cliente portarretratos.
- GifLib, biblioteca en C++ creada por Gershon Elber y adaptada para este proyecto con el fin de decodificar GIF en el cliente. Puede descargarse de la página de PAlib. El código original presenta numerosas dependencias de PAlib, una biblioteca de alto nivel para la Nintendo DS. Ya que la biblioteca no tenía las prestaciones suficientes para este proyecto, ha sido necesario incluir todo el código de escritura en pantalla de forma manual.

### B.3.2. Análisis

El estudio individual de este formato es interesante, ya que, si bien GIF nos ofrece un rango de colores limitado, sí que permite variar la paleta para obtener calidades diferentes en función de los colores, así como estudiar diferentes algoritmos de *dithering*. Este estudio llevará a cabo una comparativa de los resultados obtenidos codificando cada imagen de nuestro banco de imágenes a formato GIF con cada una de las diferentes paletas y métodos de *dithering*.

## Calidad visual

La Figura B.11 muestra la captura *tipo foto* de la Figura B.1 codificada con cada una de las diferentes paletas y con cada uno de los diferentes métodos de *dithering* para poder comparar todas las posibilidades. Además de los resultados numéricos es importante estudiar visualmente la calidad de las imágenes.

De las anteriores capturas podemos concluir que, de las tres paletas analizadas, la que produce un mejor resultado es la paleta *Quantize* en todos los casos.

Respecto al algoritmo de *dithering*, excepto el 0, que es una aproximación al color y no un algoritmo reconocido, y el 5, Stevenson-Arce, el resto de algoritmos dan resultados semejantes.

Aún así, cabe destacar que la calidad de JPEG es mayor para cualquiera de estas imágenes.

## Resultados numéricos

**Tiempo de compresión** La Figura B.12 muestra el resumen de los tiempos obtenidos en media necesarios para comprimir cada uno de los 3 tipos de imagen que hemos explicado anteriormente, teniendo en cuenta tanto las paletas como los algoritmos de *dithering*. En el eje de



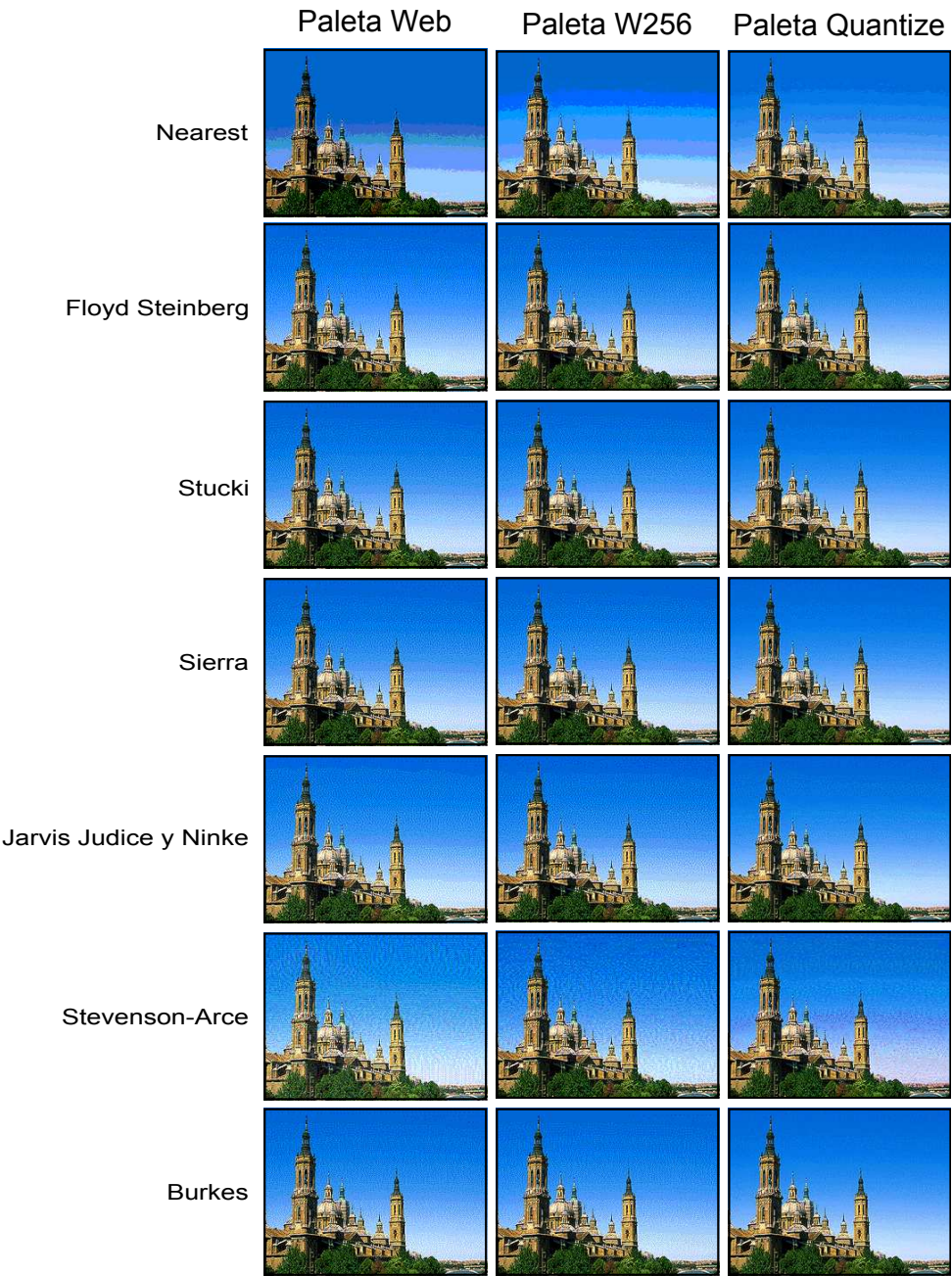
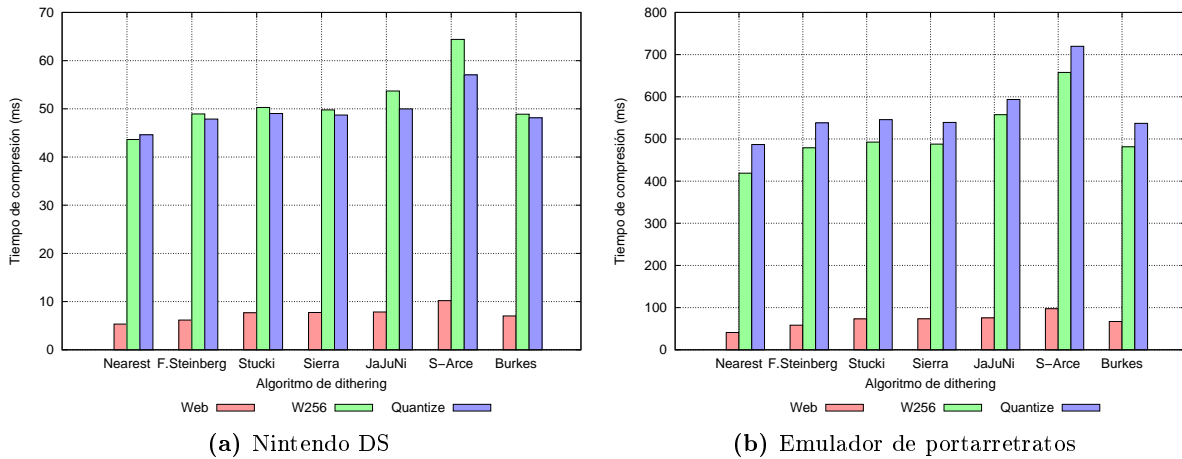


Figura B.11: Cuadro comparativo de formato GIF

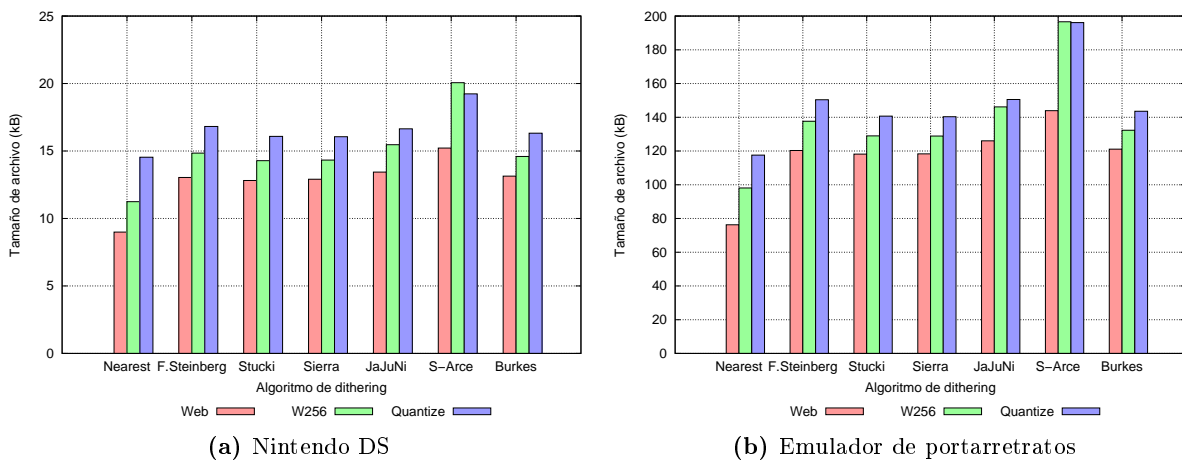
abscisas están representados los diferentes métodos de *dithering* analizados y en el eje de ordenadas se representa el tiempo de compresión en milisegundos. Las barras representan cada una de las tres paletas.



**Figura B.12:** Tiempos de compresión para formato GIF

Se ve como la paleta Web es la más rápida de todas con una amplia diferencia, mientras que las otras dos paletas, más lentas, presentan tiempos de compresión similares.

**Tamaño** La Figura B.13 muestra el resumen de los tamaños obtenidos en media de las imágenes generadas con cada uno de los tres tipos de imagen que hemos explicado anteriormente. Se tiene en cuenta tanto las paletas como los algoritmos de *dithering*. En el eje de abscisas están representados los diferentes métodos de *dithering* analizados y en el eje de ordenadas se representa el tamaño del archivo resultante en bytes. Las barras representan la media de los tamaños de archivo resultantes obtenidos para las treinta imágenes, diez de cada tipo para cada una de las tres paletas.



**Figura B.13:** Tamaño de archivo para formato GIF



De nuevo se observa cómo la paleta Web da los mejores resultados; sin embargo, la diferencia no es tan acusada como en el caso del tiempo de compresión. Las otras dos paletas también dan resultados aceptables y similares entre sí.

**Resultado global** Teniendo en cuenta los análisis anteriores, y buscando la mejor calidad para todo tipo de imágenes, se decide que la paleta óptima para este formato es la paleta Quantize. A pesar de no ser la paleta más rápida, es la que mejor calidad visual consigue. La paleta Web, a pesar de ser la más rápida, no puede ser considerada como una opción para el envío de capturas de todo tipo en nuestro proyecto, debido a su evidente falta de calidad con cualquier algoritmo de *dithering*.

Como algoritmo de *dithering* elegimos el número 2, Floyd Steinberg, ya que reúne las mejores características de los puntos estudiados.

Si se desea enviar, por ejemplo, sólo una parte de la captura anterior, como ya se ha explicado anteriormente, la elección anterior no sería viable, ya que las dos partes de la imagen tendrían diferente paleta y la imagen saldría con los colores alterados. En ese caso, se debería recurrir a una de las dos otras paletas.

De la misma manera, si consideramos sólo el modo texto, las dos paletas, tanto Web como Windows256 dan los mejores resultados. Para estos casos, elegimos la paleta Web, que los tiempos que utiliza para su compresión son mucho menores que la paleta Windows256 y la calidad en este caso es aceptable.

Respecto al algoritmo de *dithering*, elegimos nuevamente el número 2, el algoritmo de Floyd Steinberg.

De esta manera, para el formato GIF seleccionamos dos tipos de imagen:

1. Para todo tipo de capturas: Paleta Quantize con algoritmo Floyd Steinberg.
2. Para capturas con sólo texto o envíos parciales o incrementales: paleta Web con algoritmo Floyd Steinberg.

## B.4. Descripción y análisis de PNG

En la siguiente sección se lleva a cabo una descripción y un análisis detallado del formato PNG y su uso en el proyecto.

### B.4.1. Descripción

Las siglas PNG hacen referencia al formato Portable Network Graphics. El formato GIF, como se ha visto, está muy limitado por el número de colores de las imágenes. Además, aunque hoy ya está en el dominio público, en un principio GIF se encontraba cerrado bajo patentes. La creación del formato PNG buscaba reemplazar al formato GIF con un formato más adecuado a los sistemas modernos y de uso libre.

## Compresión y características

PNG utiliza compresión sin pérdida de información, al igual que GIF. La diferencia con éste, es que PNG soporta un rango mucho mayor de profundidad de color. Puede llegar a los 24 bits (8 bits por canal) o 48 bits (16 bits/canal). Otra característica de este formato es que puede estar basado en paleta o no tenerla, de allí que su calidad pueda ser mucho mayor. Además, PNG permite transparencia pero no permite animaciones.

## Uso en nuestro proyecto

Nuestro estudio va a consistir en realizar una comparativa de los diferentes tiempos de compresión y tamaño de las imágenes resultantes variando el ratio de compresión.

A pesar de utilizar un algoritmo sin pérdidas, este formato permite ajustar el ratio de compresión en un rango de 0 a 9 (máxima), comprometiendo en este caso el tamaño con el tiempo de codificación. Además permite elegir entre diferentes tipos de filtrado y el uso de entrelazado.

De estas tres opciones, tanto el filtrado como el entrelazado no presentan grandes diferencias sobre nuestro banco de pruebas, ni en tiempo ni en tamaño. Por esto, las obviaremos y nos centraremos solamente en cómo afecta el ratio de compresión a los factores de tiempo de compresión y tamaño del archivo generado para no extender el estudio.

En este caso, los resultados visuales no son importantes a la hora de tomar una decisión, ya que la calidad de la imagen es siempre perfecta. Esto es posible gracias al antes mencionado algoritmo sin pérdida y la cantidad de colores que puede representar este formato.

La Figura B.14 sirve de comparativa entre los distintos formatos. Se puede apreciar que la calidad de PNG es superior a las demás. Hay que hacer constar que la imagen original era un JPEG, típico de páginas web. Se ha preferido hacer así en lugar de partir de una imagen pura para tener un caso real de lo que el usuario va a percibir.



**Figura B.14:** Comparativa de formatos

### B.4.2. Análisis

El estudio individual de este formato pretende encontrar el ratio de compresión óptimo teniendo en cuenta el tiempo de codificación y el tamaño de imagen resultante.

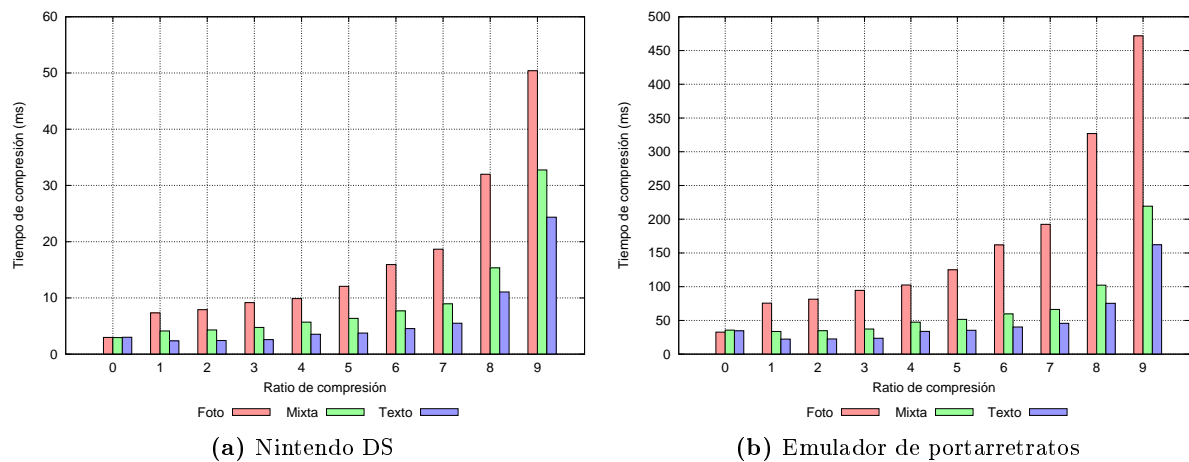
## Calidad visual

La calidad visual de las imágenes, como ya se ha dicho, es igual en todos los ratios estudiados, como se muestra en esta imagen, por lo que éste no será un factor a tener en cuenta en la decisión final.

## Resultados numéricos

A continuación se muestra el resumen de los resultados obtenidos en media para cada uno de los tres tipos de imagen que hemos explicado anteriormente, tanto para el tiempo de compresión como para el tamaño del archivo resultante.

**Tiempo de compresión** La Figura B.15 muestra cómo afecta el ratio de compresión al tiempo de comprimir el formato. En el eje de las abscisas están representados los diferentes ratios de compresión y en el eje de ordenadas el tiempo, en milisegundos, que se tarda en comprimir el conjunto de imágenes de cada tipo. Las barras representan la media de tiempo obtenido para las 10 imágenes de cada tipo.



**Figura B.15:** Tiempos de compresión para formato PNG

Se observa cómo el tiempo en crear la imagen es mayor cuanto mayor es el ratio de compresión, ya que aumenta la cantidad de trabajo a realizar. Conforme aumenta el ratio de compresión, el tiempo se incrementa de manera gradual excepto en los dos ratios mayores, con los que se produce un incremento más significativo.

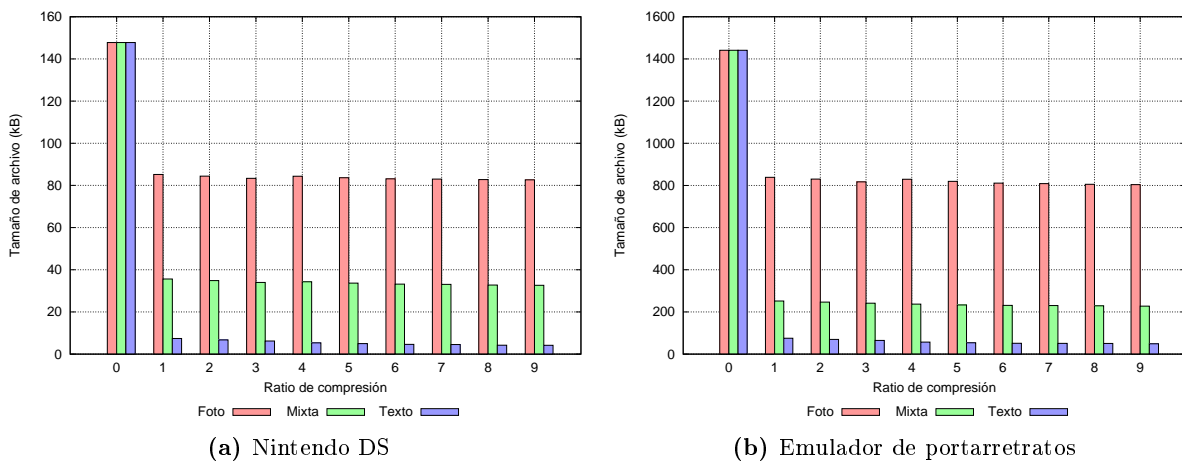
A continuación, se realiza una comparación de los diferentes formatos tomando como características para cada formato las antes obtenidas como óptimas. Para PNG se toma un ratio de compresión de 4 como óptimo.

En media, los tiempos para cada formato son los contenidos en la Tabla B.1. Como se ve con estos datos, el tiempo que utiliza PNG en comprimir es similar al que utiliza JPEG, ganando JPEG en media y PNG para imágenes con sólo texto. Puede ser interesante con estos datos usar PNG para este tipo de imágenes.

**Tabla B.1:** Comparativa de tiempo de compresión en formatos gráficos. Se indica la media en milisegundos

Tipo de imagen	PNG	JPEG	GIF
Sólo fotografía	6,3 ms	4,88 ms	49 ms
Sólo texto	3,5 ms	4,9 ms	26,4 ms

**Tamaño de archivo** La Figura B.16, muestra como afecta el ratio de compresión al tamaño final de la imagen. En el eje de las abscisas están representadas los diferentes ratios de compresión y en el eje de ordenadas el tamaño en bytes de las imágenes resultantes en media de cada tipo. Las barras muestran la media para las 10 imágenes de cada uno de estos tres conjuntos.

**Figura B.16:** Tamaño de archivo para formato PNG

Se observa cómo a excepción de la primera imagen, con compresión 0 y un tamaño muy elevado, el resto de imágenes se encuentran alrededor de 40KB por imagen con variaciones mínimas.

A continuación, se realiza una comparación de los diferentes formatos tomando como características para cada formato las antes obtenidas como óptimas.

En media, los tiempos para cada formato son los contenidos en la tabla B.2.

**Tabla B.2:** Comparativa de tamaño de archivo en formatos gráficos. Se indica la media en kilobytes

Tipo de imagen	PNG	JPEG	GIF
Sólo fotografía	41,3 KB	12,3 KB	16 KB
Sólo texto	5,3 KB	13,7 KB	4,4 KB

Como vemos, con respecto al tamaño, para imágenes con solo texto, podría ser interesante este formato junto con GIF. Sin embargo, para todo tipo de imágenes, este formato precisa de demasiado tamaño para una calidad demasiado alta para nuestras necesidades.

## Resultado global

Teniendo en cuenta los análisis anteriores, sabemos que en el aspecto visual de imágenes codificadas con cualquiera de los ratios genera imágenes de buena calidad. Respecto al tiempo, un ratio de entre 0 y 7 podría ser óptimo. Por último, el estudio del tamaño muestra que el ratio

de 0 genera un tamaño demasiado elevado, con lo cual el ratio óptimo estaría situado entre 1 y 7.

Elegimos como ratio de compresión óptimo para PNG en nuestro proyecto un ratio medio de 4. Con esta elección equilibramos el tiempo y el tamaño en un punto medio.

## B.5. Comparativa de los diferentes formatos

Con los resultados obtenidos para cada uno de los formatos estudiados, en este apartado se realiza la comparación final para efectuar la decisión de qué formato utilizar.

### B.5.1. Formatos seleccionados

Los formatos seleccionados y sus características han sido:

- JPEG. Calidad 70 %. No ha merecido la pena seleccionar ningún otro de este formato ya que no presentaba diferencias llamativas.
- PNG. Ratio de compresión 4. No ha merecido tampoco la pena seleccionar más en este formato ya que los resultados eran similares.
- GIF. Paleta Quantize y *dithering* F. Steinberg. Se ha seleccionado por su calidad frente a cualquier GIF, aunque sus tiempos son mayores que el resto de formatos.
- GIF. Paleta Web y *dithering* F. Steinberg. Se ha seleccionado por su reducido tamaño en imágenes de sólo texto. Para el resto de imágenes, su calidad no es buena.

### B.5.2. Resultados generales

A continuación se presentan los diferentes resultados, en función de los diferentes parámetros estudiados.

#### Respecto al tiempo de compresión

El estudio de la viabilidad de formatos llevado a cabo mediante la ejecución del programa con el banco de imágenes antes mencionado, arrojó los resultados respecto al tiempo necesario en transformar las imágenes a los diferentes formatos de estudio contenidos en la Figura B.17. En el eje de abscisas están los formatos de imágenes seleccionados y en el eje de ordenadas el tiempo, en media, que se tarda en comprimir la imagen expresado en milisegundos.

Se observa cómo los tiempos en media más pequeños son los obtenidos con la conversión de la imagen a formato JPEG. Aunque este formato gana en media y para imágenes con parte o todo foto, PNG es más rápido para imágenes de sólo texto.

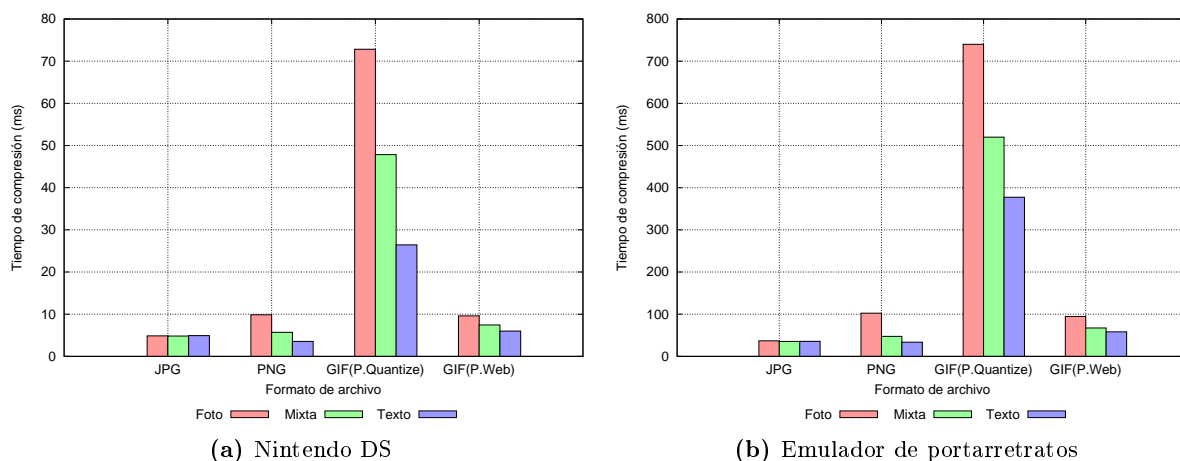


Figura B.17: Tiempos de compresión

### Respecto al tamaño de archivo

El estudio realizado arroja los resultados respecto al tamaño de archivo generados contenidos en la Figura B.18. En el eje de abscisas están los formatos de imágenes seleccionados y en el eje de ordenadas el tamaño, en media, del archivo generado para cada tipo de captura expresado en bytes.

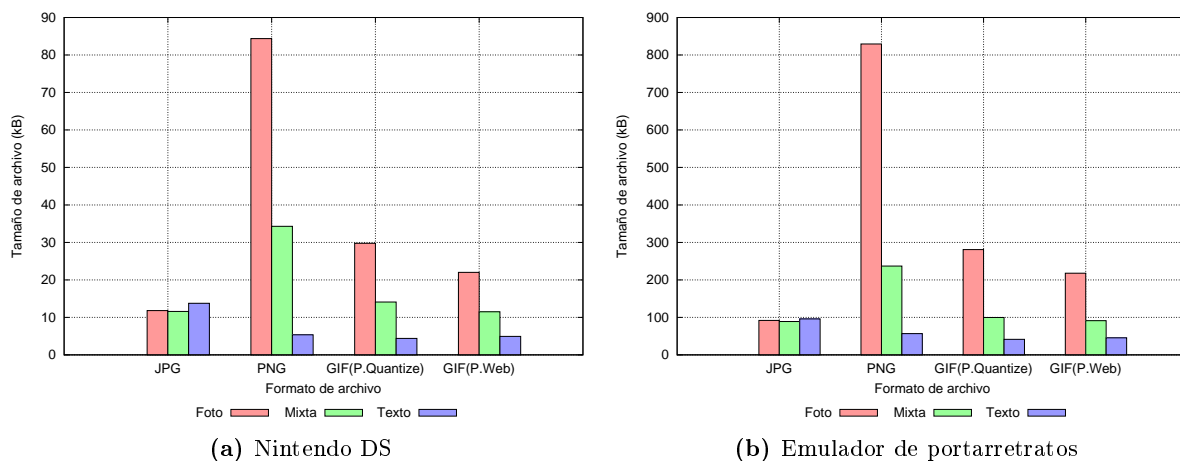


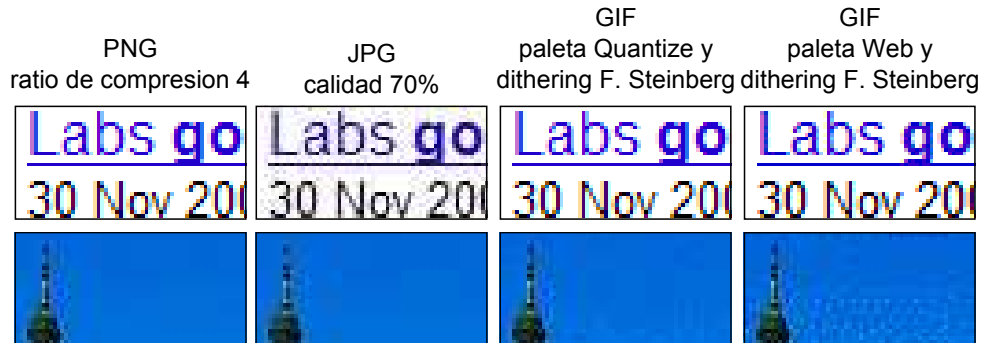
Figura B.18: Tamaños de archivo

Como se observa en la figura, la mejor opción en media es nuevamente el JPEG. Sin embargo, como en la tabla del tiempo, para las imágenes de sólo texto, los mejores resultados los encontramos en el resto de formatos, en especial el formato GIF.

### Calidad visual

Como hemos hecho con el estudio individual, es importante tener en cuenta el aspecto visual de la imagen final, ya que es lo que el cliente va a percibir de la solución.

La Figura B.19 muestra capturas ampliadas de dos tipos de imagen para una mejor comparación de los formatos seleccionados. Se observa cómo las calidades son diferentes según el tipo de imagen.



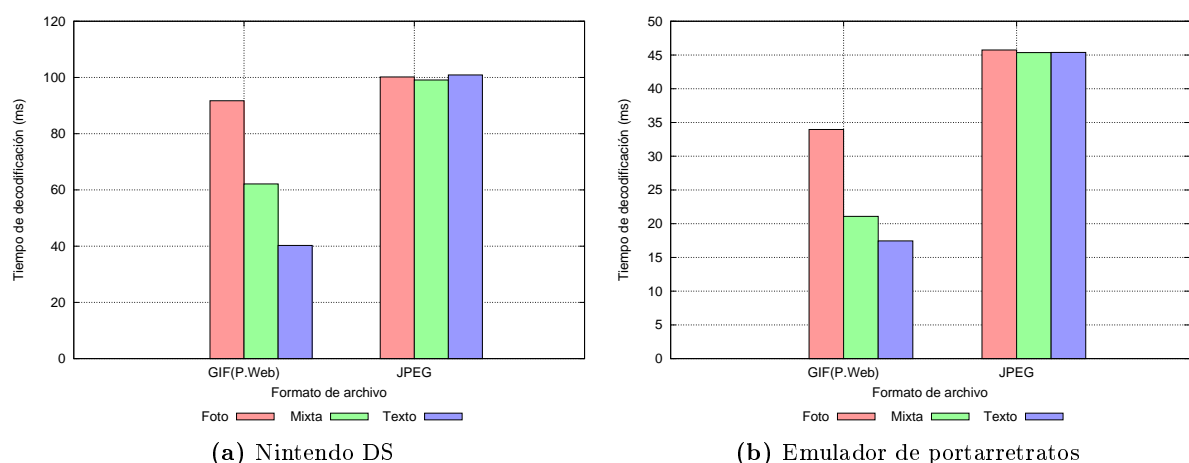
**Figura B.19:** Comparativa de los formatos seleccionados

Para imágenes de sólo texto, además de PNG que siempre presenta una buena calidad, los dos formatos GIF son superiores al formato JPEG.

Para imágenes con foto, PNG tiene la mayor calidad, seguido de JPEG que tiene una calidad más que aceptable. Los formatos GIF no resultan eficientes para este tipo de imágenes.

### Tiempo de decodificación

El tiempo de decodificar cada uno de los formatos anteriores está representado en la Figura B.20. Este dato no se ha estudiado como dato decisivo en esta comparación, sino como apunte final para hacer una idea de los tiempos totales que conlleva el envío de las capturas.



**Figura B.20:** Tiempos de decodificación

Se observa cómo el tiempo en media empleado para decodificar cualquier tipo de imagen es similar en ambos formatos, siendo el formato GIF ligeramente mejor. Esta mejoría la consigue sobretodo con el tiempo empleado en decodificar imágenes de sólo texto, consiguiendo así reducir

su media.

Como se puede comprobar en la Figura B.20, los resultados son proporcionales para cada una de las máquinas estudiadas.

### B.5.3. Valoración final

Si tenemos en cuenta tanto los resultados del tiempo de compresión como los obtenidos midiendo el tamaño de la imagen final, para cualquier tipo de imagen, vemos como el formato JPEG reúne los mejores datos. Por otra parte, hemos comprobado el aspecto visual de las diferentes conversiones de formatos, viendo que el formato JPEG da un resultado más que satisfactorio para todo tipo de imágenes.

Para realizar las pruebas llevadas a cabo para estudiar la viabilidad de los formatos, se han implementado cada una de las opciones que hemos presentado en este capítulo. Como se desprende de los datos finales del estudio, no todas han dado resultados satisfactorios, así que hemos decidido dejar implementadas solamente las mejores de ellas. Las dos opciones elegidas han sido:

- **Formato JPEG**

Se deja como opción por defecto y con la calidad ajustable por el usuario según sus necesidades pudiendo seleccionar una calidad diferente para cada pantalla.

- **Formato GIF**

Se deja implementado con paleta Web, que puede ser útil para las imágenes de solo texto a decisión del usuario.

De las optimizaciones presentadas en la Sección 4.2.3 de la memoria, la única que realmente mejora la comunicación, como ya hemos visto en esa misma sección, es la de los envíos de secciones de imagen debido a los desplazamientos. Por este motivo, esta optimización se ha dejado implementada en la aplicación.



## Anexo C

# Manual de la aplicación

---

Este anexo recoge los manuales de usuario de las aplicaciones implementadas.

### C.1. Proxy

El diseño y la implementación del proxy se han descrito en detalle en el Capítulo 2 y el Capítulo 4. A continuación se presentan las características principales del producto software resultante, así como unas breves instrucciones de uso.

#### C.1.1. Entorno operativo

El proxy se ejecuta sobre cualquier computadora con una instalación estándar de Windows XP. Es necesario que esté previamente instalado Internet Explorer, con los complementos que se consideren oportunos para la consulta de páginas web.

El proxy actuará de puente entre el cliente y los contenidos a los que desea acceder. Por tanto, será necesario que incorpore una conexión de red, a través de la cual se comunicará con el cliente. Por norma general, el proxy dispondrá de acceso a Internet, desde la propia red en la que está el cliente o desde cualquier otra.

#### C.1.2. Puesta en marcha

La aplicación proxy que debe residir en el PC no precisa de instalación previa, basta con ejecutar el fichero del programa para iniciarlo y poder empezar a trabajar con él.

#### C.1.3. Funcionamiento

Al arrancar la aplicación nos encontramos con la pantalla representada en la Figura C.1. En dicha pantalla tenemos diferentes configuraciones modificables por el usuario.

En la parte izquierda de la pantalla se encuentra un cuadro de texto en el cual se mostrarán mensajes derivados del uso de la aplicación. Ejemplos de estos mensajes pueden ser fallos de autenticación o información de las conexiones de dispositivos.

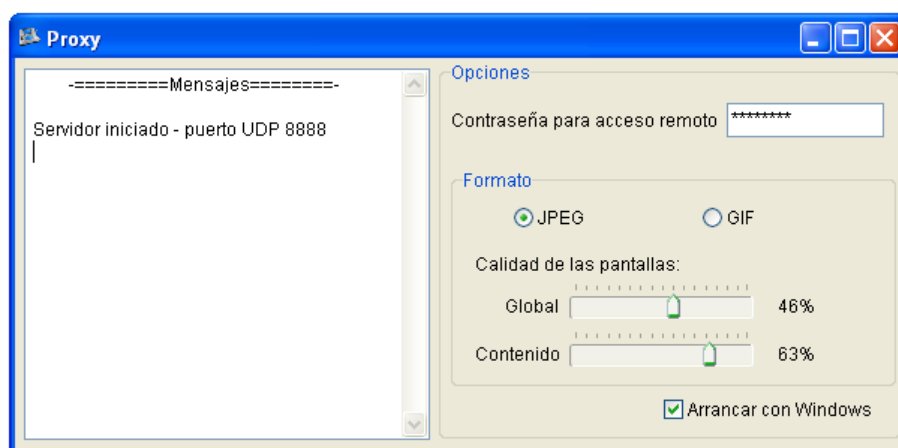


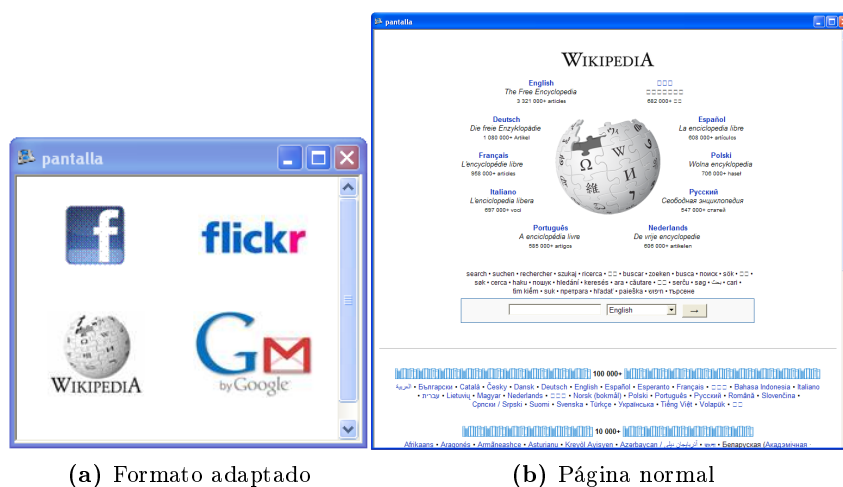
Figura C.1: Pantalla del proxy

En la parte derecha se encuentra la zona de configuración. En primer lugar está la posibilidad de poner una contraseña para evitar la entrada al PC cuando se pretende acceder al escritorio remoto. Puede dejarse en blanco si es preciso.

A continuación, se encuentra la configuración del formato, a elegir entre GIF y JPEG. En el caso de elegir JPEG se presentan dos barras mediante las cuales el usuario puede determinar la calidad que precisa para cada una de las pantallas. En el caso de elegir GIF, la configuración está predefinida con la paleta Web.

Por último encontramos la opción de “Arrancar con Windows”. Esto resulta de utilidad si usamos mucho la aplicación o queremos que al iniciar el ordenador el proxy ya este preparado para recibir clientes.

Además de esta pantalla, el proxy lanza navegadores de acuerdo con el contenido que desea ver el usuario. Las ventanas de navegación serán de tamaño reducido si se desea ver contenido adaptado para la resolución del dispositivo, y de un tamaño superior si se desea visualizar una página web completa. La Figura C.2 muestra posibles formas del navegador.



(a) Formato adaptado

(b) Página normal

Figura C.2: Formatos del navegador del proxy

## C.2. Cliente para Nintendo DS

Se ha realizado una implementación de referencia del dispositivo cliente utilizando la consola Nintendo DS. A continuación se describe el funcionamiento de dicho cliente.

### C.2.1. Puesta en marcha

Para iniciar la aplicación en el dispositivo cliente elegido, se precisa cargar el ejecutable en una tarjeta de memoria, la cual estará contenida en el interior del dispositivo de desarrollo de forma análoga a un juego de la consola. Una vez dentro del menú de la propia herramienta, se ha de seleccionar “Portarretratos”, iniciando así la aplicación.

### C.2.2. Funcionamiento

Al iniciar la aplicación, aparece el menú representado en la Figura C.3, en el cual se nos da a elegir el método de conexión. Las opciones son las siguientes:

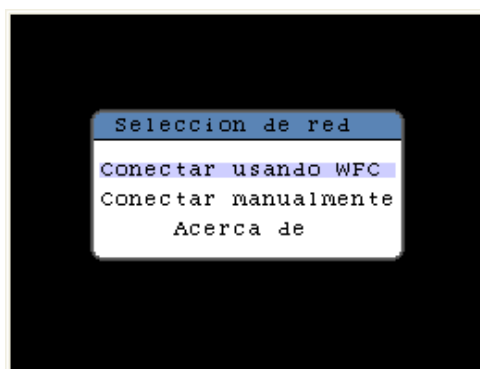


Figura C.3: NDS. Menú de conexión

- **Conectar usando WFC**

Si se selecciona esta opción, la máquina intentará conectarse a través de WFC, la conexión Wi-Fi de Nintendo, o lo que es lo mismo, a través de una conexión ya establecida anteriormente, cuyos datos se guardan en la memoria de la consola.

- **Conectar manualmente**

Si se selecciona esta opción, el usuario deberá realizar todos los pasos necesarios para autenticarse en la red. La siguiente pantalla mostrará los puntos de acceso encontrados por la consola, representado en la Figura C.4.

Una vez cargados, se selecciona el punto de acceso al cual se quiere conectar y se le solicita la clave como se muestra en la Figura C.5.

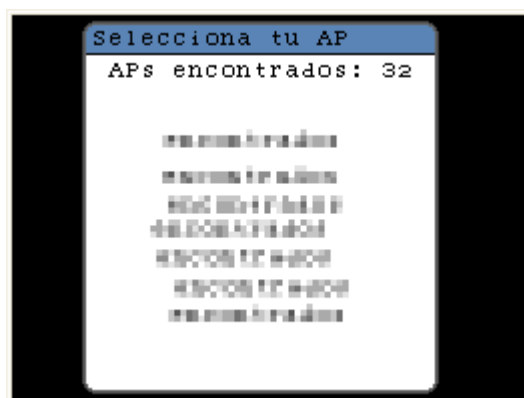


Figura C.4: NDS. Selección de AP



Figura C.5: NDS. Petición de contraseña

En cualquiera de los dos casos, si la conexión tiene éxito, a continuación se solicitará la IP del PC en el que se ejecuta el proxy, pantalla que se puede ver en la Figura C.6.

Una vez aceptada la clave de red, se llega a la pantalla general, representada en la Figura C.7, en la cual se puede elegir qué contenido visualizar. Según la opción elegida accederemos a contenidos diferentes.

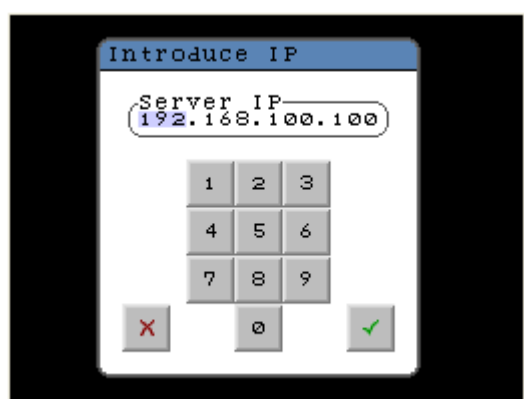


Figura C.6: NDS. Solicitud de IP

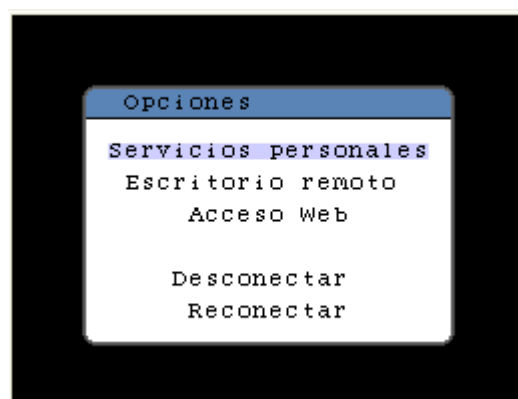


Figura C.7: NDS. Menú de la aplicación

#### ■ Servicios personales

Mediante esta opción, se accede a los contenidos previamente solicitados en la web para ser visualizados, ya preparados para la pantalla de la consola. La Figura C.8 muestra el modo

de visionado de los servicios solicitados.



**Figura C.8:** NDS. Servicios personales

#### ■ Escritorio remoto

Mediante esta opción, se accede a visualizar el escritorio del PC en el que se está ejecutando el proxy. Con el fin de controlar el acceso a esta información, se solicita una clave que deberá haber sido puesta en el proxy anteriormente. La pantalla que nos encontramos es la representada en la Figura C.5.

Una vez la contraseña es aceptada, se visualiza el escritorio tal y como se muestra en la Figura C.9.

#### ■ Acceso Web

Seleccionando esta opción del menú se accede a la navegación de páginas web en formato normal. La Figura C.10 muestra cómo se visualiza este contenido en la consola. En la pantalla de arriba se visualiza la pantalla completa mientras que en la de abajo se encuentra una ampliación de la parte superior.

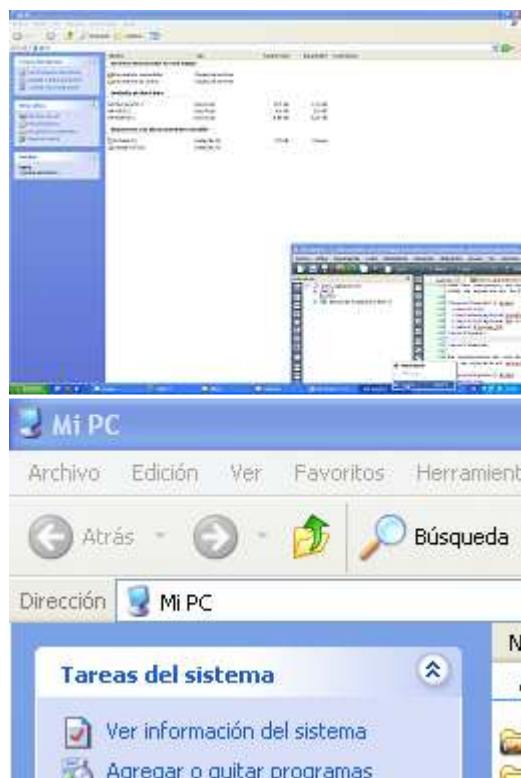


Figura C.9: NDS. Escritorio remoto



Figura C.10: NDS. Acceso Web

### ■ Desconexión

Esta opción permite al usuario desconectarse del proxy y volver a la pantalla inicial, representada en la Figura C.3.

### ■ Reconexión

Esta opción nos reconecta de nuevo al servidor si ha habido algún problema.

## C.2.3. Otras funcionalidades

En cualquier momento de la ejecución del programa se puede acceder a una pantalla de opciones representada en la Figura C.11 mediante el botón *Start*.

En esta pantalla se puede definir si el usuario es zurdo o diestro, así como la velocidad de refresco de las imágenes. También se puede volver al menú principal de la Figura C.7.

Además, es posible en todo momento sacar un teclado pulsando la tecla *Select*, como el mostrado en la Figura C.12.



Figura C.11: NDS. Menú de opciones

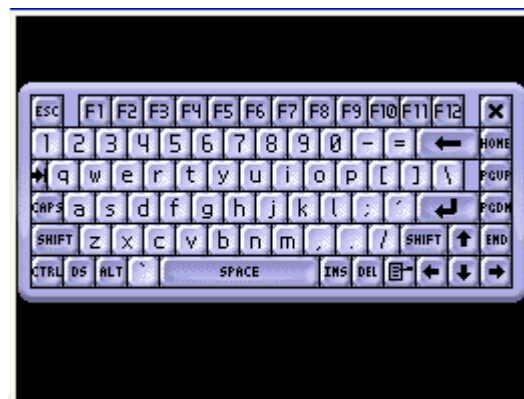


Figura C.12: NDS. Teclado

### C.2.4. Teclado

A continuación se detallan las funciones de cada botón de la consola Nintendo DS. Como puede verse en la Figura C.13, para la opción de escritorio remoto no son necesarias todas las teclas.



Figura C.13: Teclas DS

Pulsando la tecla “L” mientras desplazamos el lápiz sobre la pantalla táctil, permite desplazar la pantalla tanto vertical como horizontalmente.

La tecla “invertir pantallas” Puede resultar útil si queremos interactuar con la pantalla de arriba, ya que únicamente la pantalla inferior es táctil. Gracias a esta opción es posible intercambiar el contenido de ambas pantallas.

### C.3. Cliente para emulador de portarretratos

Se ha realizado una segunda implementación de un dispositivo cliente utilizando un ordenador portátil. Tan sólo se hace uso de la pantalla, el teclado, el ratón y la conexión a Internet. A continuación se describe el funcionamiento de dicho cliente.

#### C.3.1. Puesta en marcha

La aplicación que emula el portarretratos no requiere instalación previa. Para su uso, basta con ejecutar el fichero del programa.

#### C.3.2. Funcionamiento

El funcionamiento es sencillo. La conexión a Internet viene dada por el portátil en este caso, así que no se requiere configuración de red.

Al arrancar el programa se nos muestra un menú en el cual se nos da a elegir entre diferentes resoluciones de pantalla. Al tratarse de un emulador, se han seleccionada resoluciones típicas de marcos digitales. Una vez elegida la resolución, se muestra un menú con las opciones de navegación disponibles. Ya que la pantalla es mayor que en el caso de la Nintendo DS y no es tan necesario el contenido adaptado, sólo se presentan dos opciones:

- **Navegación web**

Mediante esta opción se accede a la navegación de páginas web normales, servidas a través del proxy.

- **Escritorio remoto**

Esta opción permite controlar el PC en el cual se esté ejecutando el proxy.

#### C.3.3. Teclado

El método de interactuar con las capturas se hace a través del ratón y el teclado. Las teclas a utilizar son las representadas en la Figura C.14.

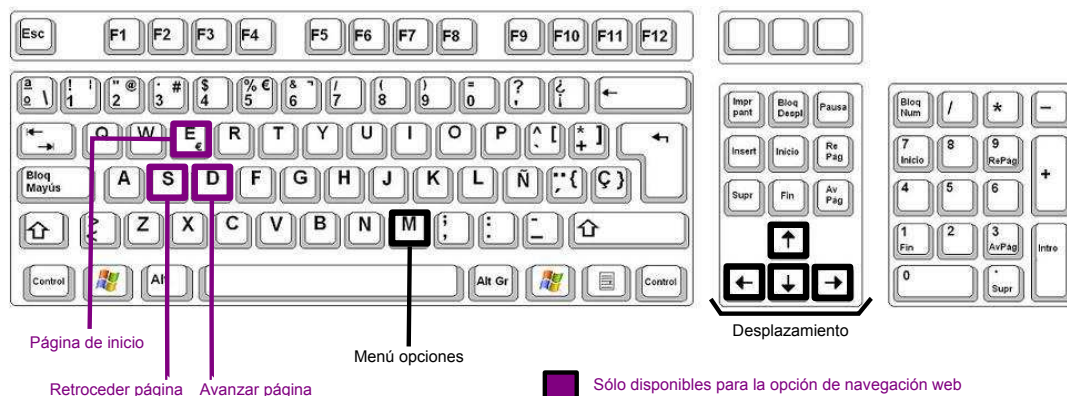


Figura C.14: Teclado del emulador de portarretratos



## C.4. Servidor

El servidor incluye una base de datos con toda la información relativa a las preferencias y contraseñas de los clientes. A su vez, cuenta con páginas con contenidos adaptados, preparadas para ser visualizadas en dispositivos con diferentes resoluciones de pantalla.

Las páginas adaptadas, preparadas para su visualización en la DS, forman parte del servidor. El proxy accede a ellas cuando son solicitadas por el cliente. Un ejemplo de estas páginas son las presentadas en la Figura C.8

El usuario no tiene contacto directo con el servidor en el transcurso de la ejecución de la aplicación, aunque sí puede programar páginas personalizadas e incluirlas en el servidor si así lo desea.



## Anexo D

# Protocolo de comunicaciones proxy-cliente

---

En este anexo se recoge toda la información relativa a la comunicación entre el proxy y el cliente de la solución.

### D.1. Introducción

La comunicación entre el proxy y el cliente se realiza siguiendo un protocolo de comunicaciones, definido durante este proyecto. Con el fin de facilitar el diseño de nuevos clientes, hemos documentado la especificación completa del protocolo.

### D.2. Protocolo de comunicación

El protocolo, cuya representación gráfica puede consultarse en la Figura D.1, funciona de la siguiente manera para los envíos de las capturas:

1. El cliente solicita una nueva imagen. Para esto le envía una trama.
2. El proxy le envía una trama informándole de que ha recibido la petición y le especifica el número de bloques en los que se divide la imagen que le enviará.
3. El cliente guarda sitio para recibir la imagen completa y la inicializa. Además, envía una trama al proxy para informarle de que ya puede empezar a enviar los bloques de la imagen.
4. El proxy, envía las diferentes partes que componen la imagen seguidos mediante tramas. Cada una de ellas tiene el número de secuencia para poder reconstruirla en recepción.
5. Si hay algún bloque que no se ha recibido, o llega con error, se envía inmediatamente un NAK con el número de trama que le falta al proxy.
6. El proxy reenviaría una trama con el bloque perdido o dañado.
7. Cuando recibe todas las tramas, se envía un ACK para informar al proxy de que todos los bloques de la imagen han sido recibidos.

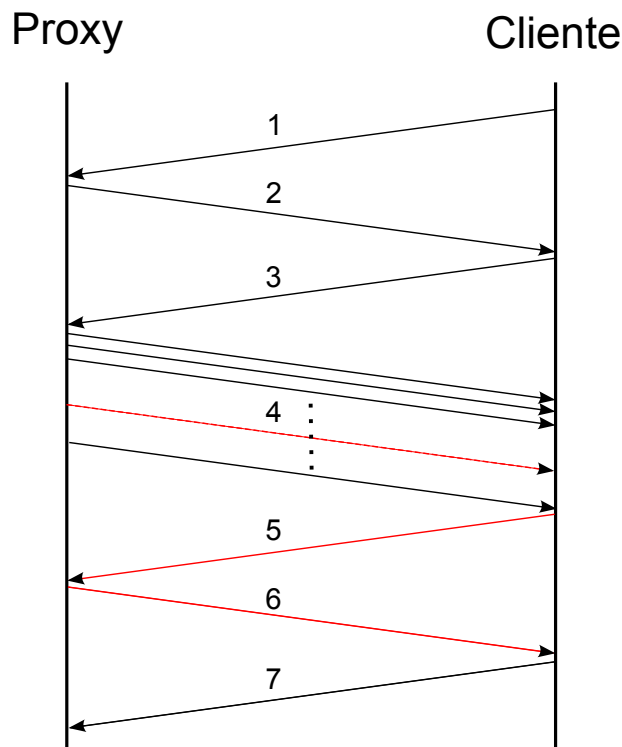


Figura D.1: Protocolo proxy-cliente

### D.3. Formato de tramas

Cada una de las tramas enviadas durante la comunicación proxy-cliente cuenta con un código inicial, que permite identificar el tipo de trama. Los campos que contiene cada trama están prefijados en función de su tipo.

La Tabla D.1 y la Tabla D.2 muestran los diferentes tipos de trama y sus campos correspondientes. Si no se especifica lo contrario, cada campo ocupa exactamente un *byte*. En la siguiente sección se explicará qué acciones se llevan a cabo por el cliente y el proxy cuando reciben estas tramas.

### D.4. Interpretación de las tramas

A continuación se detalla cómo debe interpretar las tramas cada uno de los elementos de la arquitectura.

#### D.4.1. Tramas recibidas por el proxy

Las diferentes tramas que pueden ser recibidas por el proxy se resumen en la Tabla D.1. La descripción detallada de su significado y las acciones derivadas de su recepción, atendiendo a la numeración de la tabla, se presentan a continuación.

1. Indica que un usuario quiere conectarse. El segundo campo de la trama contiene la contraseña mediante la cual se quiere conectar. El proxy deberá hacer la petición al servidor,

**Tabla D.1:** Tipos de trama enviados por el cliente

Num	Tipo Trama	Campo		
		1	2	3
1	USUARIO	password (n Bytes)		
2	CONNECT	password (n Bytes)		
3	DISCONNECT			
4	PING			
5	MOUSECOORD	x	y	
6	MOUSECOORDZOOM	x	y	
7	MOUSECLICKDOWN			
8	MOUSECLICKUP			
9	KEYPRESS	key		
10	CAPTURA			zoom
11	CAPTURAPEQ	x	y	zoom
12	RECIBIDO	CAPTURA/ CAPTURAPEQ		
13	OK	CAPTURADATA/ CAPTURAPEQDATA	bloques	
14	NAK	CAPTURADATA/ CAPTURAPEQDATA	n.bloque	
15	TECLA1			
16	TECLA2			
17	TECLA3			
18	MODO	x		

que es el que tiene la base de datos con la información de los usuarios y lleva a cabo la autenticación.

2. Indica que un usuario quiere acceder al escritorio remoto del PC en el que se ejecuta el proxy. La contraseña se compara con la establecida en el propio proxy. A continuación, se devuelve un mensaje para indicar el resultado de la operación.
3. Indica que el usuario se ha desconectado voluntariamente, así que se procede a liberar sus recursos.
4. Es una trama de control. El proxy espera recibir del cliente este tipo de tramas dentro de un tiempo establecido. Sirve para saber si un usuario se ha desconectado. Si no se recibe esta trama en dicho tiempo, se le desconecta del servicio.
5. Permite dibujar el ratón de la pantalla global, gracias a las coordenadas recibidas.
6. Permite dibujar el ratón en la pantalla de contenido, así como localizar los puntos en los que se ha hecho un clic de ratón.
7. Permite realizar un clic de ratón.
8. Permite saber cuándo se libera el ratón después de haber sido pulsado.
9. Indica que una tecla ha sido pulsada y cuál ha sido. El proxy, cuando recibe esta trama, realiza la pulsación de la misma tecla en su navegador o su escritorio.
10. Consiste en una petición por parte del cliente de una nueva captura para la pantalla global (CAPTURA). El proxy, al recibir esta trama, capturará la pantalla correspondiente y le enviará al cliente otra trama para informarle sobre el número de bloques de la nueva captura, además de otros datos de la misma. En caso de que la captura no haya variado con la anterior, no se enviará nada al cliente.
11. Es similar a la anterior pero referida a la pantalla de contenido(CAPTURAPEQ).
12. Informa al proxy de que la captura que se estaba enviando ha sido recibida en su totalidad.
13. Informa al proxy de que el cliente está preparado para recibir la imagen. El segundo campo CAPTURA/CAPTURAPEQ le informa de cuál es la pantalla que requiere la captura. A continuación, enviará las partes que componen la captura a la pantalla que lo ha solicitado.
14. Si el proxy recibe esta trama, sabe que una trama de las que ha enviado no ha sido recibida o ha llegado dañada. Inmediatamente, reenviará la trama cuyo número corresponde con el campo n.Bloque.
15. El proxy recibe que se ha pulsado la tecla 1 del cliente y, según el servicio que estemos mostrando, realizará una función u otra. En nuestro caso, si el cliente se encuentra en modo de navegación normal, se cambiará a la página anterior del historial de navegación.
16. El proxy recibe que se ha pulsado la tecla 2 del cliente y, según el servicio que estemos mostrando, realizará una función u otra. En nuestro caso, si el cliente se encuentra en modo de navegación normal, se cambiará a la página siguiente del historial de navegación.
17. El proxy recibe que se ha pulsado la tecla 3 del cliente y, según el servicio que estemos mostrando, realizará una función u otra. En nuestro caso, si el cliente se encuentra en modo de navegación normal, se cambiará a la página de inicio del navegador.

18. El proxy gracias a esta trama sabe qué modo del menú ha elegido el usuario, con el fin de saber de dónde debe obtener las capturas y qué teclas corresponden con ese modo: 0 para el escritorio remoto y 1 para la navegación web.

#### D.4.2. Tramas recibidas por el cliente

La Tabla D.2 muestra las diferentes tramas que puede recibir el cliente. A continuación se explican las acciones que lleva a cabo el cliente en su recepción, atendiendo a la numeración de la tabla.

**Tabla D.2:** Tipos de trama enviados por el proxy

Num	Tipo Trama	Campo				
		1	2	3	4	5
1	USUARIO	booleano				
2	CONNECT	booleano				
3	CAPTURA/ CAPTURAPEQ	bloques	DIM	TAM	MOV	TIPO
4	CAPTURADATA/ CAPTURAPEQDATA	n.Bloque	datos (1KB)			

1. Esta trama le llega al cliente después de haber solicitado la conexión. El segundo campo será 0 ó 1 según su conexión haya sido o no aceptada. En caso de no serlo, se volverá a pedir la contraseña.
2. Esta trama llega después de que el cliente pretenda acceder al escritorio remoto del ordenador sobre el que se ejecuta el proxy. El campo booleano de esta trama indica si la contraseña ha sido aceptada o no.
3. Cuando el cliente recibe esta trama, inicializa la captura que recibirá para la pantalla global (CAPTURA) o para la de contenido (CAPTURAPEQ). Para ello, pondrá sus datos a cero y reservará espacio en memoria para almacenar las partes que la componen.  
El campo TIPO permite conocer qué formato de imagen posee la imagen que se va a recibir con el fin de decodificarla con una u otra librería (GIF o JPEG). El campo MOV informa si la captura ocupa toda la pantalla o, por el contrario, es una sección de la misma. El campo TAM contiene el tamaño de la imagen a enviar, ya que, de nuevo, puede ser el tamaño completo o sólo una sección. El campo DIM indica la posición en la que debe dibujarse la imagen enviada. En el caso de ser una imagen completa, este campo no se utiliza.
4. Cuando el cliente recibe esta trama, guarda a partir de la posición de memoria reservada para ese número de bloque los datos recibidos. De esta manera, cuando todos los trozos se hayan recibido, se obtendrá la imagen completa. Como sabemos que cada bloque es de tamaño fijo (ocupa exactamente 1000 bytes), es sencillo almacenar la imagen por bloques, ya que cada bloque comenzará en una posición de memoria fija.