Escuela de
Ingeniería y Arquitectura
Universidad Zaragoza

Driver universal para impresora / etiquetadora appl/SILO

Patricia Robles Alfranca

Director: Chema Guallar Leza (HP)
Ponente: Pedro Álvarez Pérez-Aradros

Proyecto Fin de Carrera
Departamento de Informática e Ingeniería de Sistemas
Escuela de Ingeniería y Arquitectura
Universidad de Zaragoza
Febrero 2012

Driver universal para impresora / etiquetadora appl SILO

RESUMEN

Este proyecto fin de carrera comprende el análisis, diseño e implementación de un driver universal basado en la tecnología XML para facilitar la impresión de documentos y etiquetas que genera la aplicación SILO de la empresa *Hewlett-Packard* (HP).

SILO es un producto creado por el departamento de Logística de HP para facilitar la gestión, de forma integrada, de todas las operaciones que se llevan a cabo en un almacén. El sistema que aquí se describe ha sido desarrollado con el fin de ser integrado en la aplicación SILO y ofrecer una solución a algunas de sus carencias.

El sistema tiene dos funcionalidades básicas. Por un lado, permite la creación de plantillas, que sirve para guardar la configuración del tipo de documento que se desea generar. Los usuarios podrán diseñar sus plantillas y guardarlas para poder disponer de ellas cuando sea necesario. Se podrán ver y editar las plantillas, modificando el formato o la disposición de sus elementos e incluso duplicarlas, generando una nueva plantilla a partir de la original y guardando ambas. Por otro lado, se encarga de gestionar la impresión de documentos, reportes o etiquetas, que puede generar la aplicación SILO, mediante un driver unificado. Este driver permite imprimir en diferentes dispositivos de distintos fabricantes, sin necesidad de tener que instalar un driver específico para cada uno de ellos. Al igual que cualquier driver, permite aprovechar las capacidades del dispositivo de impresión y ofrece al usuario diferentes opciones para gestionarlas. Otra funcionalidad añadida es la de permitir a los administradores de los dispositivos modificar su configuración. Al ser un sistema desarrollado para un entorno multiusuario y multitarea, es capaz de atender las peticiones de impresión recibidas por parte de distintos usuarios y aplicaciones de manera simultánea. Para almacenar toda la información relativa a plantillas, documentos, dispositivos y peticiones se dispone de la base de datos SILO. De esta forma todos los componentes de la aplicación SILO pueden acceder a la información almacenada.

El sistema consta de dos aplicaciones:

- Una aplicación *front-end*, desarrollada en Java, que permite al usuario la generación de las plantillas y la impresión de documentos, así como la configuración de las opciones de impresión de los dispositivos.
- Una aplicación *motor*, desarrollada en C, que se encarga de atender las peticiones de impresión que provienen tanto de la aplicación *front-end*, como de otras aplicaciones.

La tecnología principal utilizada es XML, que es la que permite la generación de documentos y, junto con el motor de transformación XSL, su traducción al lenguaje nativo de impresión del dispositivo. Esta forma de automatizar la traducción de los documentos es una mejora añadida al conjunto de servicios que ofrece HP a sus clientes para la gestión de sus almacenes.

Índice general

Índice de figuras

Capítulo 1

Introducción

En este primer capítulo se introduce de forma general el proyecto. En la sección 1.1 se describe el contexto de realización del mismo. En la sección 1.2 se relata el problema real que conduce a la búsqueda de una solución en forma de proyecto. A continuación en la sección 1.3 se detallan las tecnologías utilizadas durante el proyecto y, por último, la sección 1.4 describe la estructura de esta memoria para dar una visión general sobre su organización.

1.1 Contexto del proyecto

El desarrollo de este proyecto se enmarca en el plan de estudios de Ingeniería Informática de la Escuela de Ingeniería y Arquitectura de la Universidad de Zaragoza. Surge a partir de un acuerdo de colaboración entre el BIFI (Instituto de Biocomputación y Física), en la Universidad de Zaragoza, y la empresa HP (Hewlett-Packard) con sede en Zaragoza.

HP es una gran empresa que opera en varios países repartidos por el mundo. Aquí, en España, tiene su sede principal en Madrid y en los últimos años, se han implantado nuevas sedes en otros puntos del país, incluyendo Zaragoza. Esta compañía ofrece una amplia gama de servicios y productos tecnológicos relacionados con la información y la comunicación. Sus servicios están destinados tanto a empresas - pequeñas, medianas o grandes- como a particulares y van desde entretenimiento digital e impresión doméstica hasta computación.

Uno de sus productos es SILO (*System for Integrated LOgistic*). Según la traducción de sus siglas en inglés, SILO, es un sistema diseñado para integrar los diferentes servicios que proporciona a sus clientes el departamento de Logística de HP. Es una herramienta cuya función es gestionar, de forma integrada, todas las operaciones que se llevan a cabo en un almacén. Para ello se basa en un sistema ERP (*Enterprise Resource Planning*), que suele utilizarse para manejar la producción, logística, distribución, inventario, envíos, facturas y contabilidad de una compañía. También puede intervenir en el control de las distintas actividades de negocio como ventas, entregas, pagos, producción, administración de inventarios, calidad de administración y la administración de recursos humanos. SILO consta de una aplicación Web en la que ofrece todos estos servicios a los clientes, atendiendo sus peticiones en tiempo real desde cualquier localización. Todas las peticiones de los clientes son centralizadas y según las acciones a realizar se les asigna una prioridad de servicio. Debajo de esta capa

de presentación, se encuentra el software que proporciona estos servicios y en el que se integrará la solución diseñada para este proyecto.

Actualmente la aplicación carece de un driver unificado de impresión para los distintos reportes y etiquetas que puede generar la aplicación. En un almacén se crean continuamente nuevas etiquetas que pueden ser impresas en uno o varios dispositivos. Esto obliga a escribir código particular asociado a cada nueva etiqueta generada, dependiendo del dispositivo en el que se va a imprimir. Esto mismo se aplica también a otro tipo de documentos que son generados con frecuencia como facturas, albaranes o informes.

1.2 Problema

Hoy en día, cualquier compañía dispone de diferentes modelos de dispositivos, desde impresoras o etiquetadoras hasta terminales de punto de venta en establecimientos comerciales. En una empresa de logística tiene una gran importancia la generación etiquetas de distintos tamaños para la identificación de los productos que transportan. La impresión de etiquetas es una operación muy repetida en un almacén y por ello no resulta nada eficiente el tener un driver adecuado a cada dispositivo. Al no existir un driver genérico para todos los dispositivos existentes en una empresa, la gestión de cada uno de ellos se realiza a través de sus drivers específicos. El principal problema que presenta la aplicación SILO de HP es que la generación de nuevas etiquetas o modificación de las ya existentes implica la creación de un driver específico que servirá sólo para dispositivos de un mismo fabricante. Cualquier cambio que se efectúe sobre la etiqueta implicará reescribir de nuevo su driver. Del mismo modo, si se quisiera imprimir en un dispositivo perteneciente a otro fabricante, cuyo lenguaje nativo fuera diferente, habría que crear un nuevo driver para dicha etiqueta.

La solución para evitar tener que instalar un driver concreto para cada dispositivo sería la aplicación de uno que fuera universal y se encargara de gestionar todos los dispositivos físicos dentro de una organización. Este tipo de controlador haría más sencilla la instalación y gestión de los dispositivos de impresión para los usuarios finales. Se crearía la oportunidad de tener una mejor visión del entorno de impresión de toda la organización, donde poder agrupar juntos los dispositivos para una gestión más centralizada. Por otro lado, se debe completar la solución a este problema con una aplicación que sea capaz de generar las diferentes etiquetas o documentos y guardar su formato a modo de plantilla.

El objetivo del proyecto es conseguir un driver/traductor genérico multiplataforma que permita aprovechar las características de cualquier impresora/etiquetadora, sin importar su lenguaje nativo (ZPL, PGL, PCL,..), a través de un driver genérico XML/XSL. Su finalidad es formar parte de uno de los módulos de la aplicación SILO de HP para añadir una nueva funcionalidad al conjunto de servicios que ofrecen a sus clientes para la gestión de sus almacenes.

1.3 Tecnologías y herramientas

Esta sección presenta todas las tecnologías y herramientas que han servido como soporte para elaborar este proyecto. Para su resolución se han utilizado las siguientes herramientas, clasificadas según el papel que desempeñan.

- Herramientas de análisis y diseño:
  ArgoUML v0.24
  Microsoft Visio
  Edge Diagrammer

- Herramientas y lenguajes de desarrollo:
  Máquina virtual Oracle VirtualBox
  Linux Red Hat Enterprise 5
  Oracle Server
  JDK 6.25
  IDE: NetBeans 6.0.1
  Debugger de Linux.
  Lenguaje Java[10], C y PL/SQL[11].
  XML[4][5] y XSL[7].
  Lenguajes de comandos de impresión: PCL[14][15], ZPL[16], PGL[17]

- Herramientas de planificación:
  Microsoft Project 2010

- Herramientas de prueba:
  Impresoras de prueba: HP y Zebra
  Código demo de HP.

- Herramientas de edición:
  Office 2010
  Notepad++

1.4 Estructura de la memoria

La memoria se divide en cinco capítulos, que a su vez se subdividen en varias secciones según su contenido. Este primer capítulo es simplemente una pequeña introducción al proyecto. El resto de capítulos describen de forma más profunda cómo ha sido el desarrollo del proyecto.

- El capítulo 2 describe el análisis del problema a resolver. Primero enumera los objetivos del proyecto de forma general y a continuación desarrolla esos objetivos de manera más extensa.

- El capítulo 3 presenta el diseño de la solución elegida para resolver el problema planteado en los capítulos anteriores. Primero se describe la arquitectura del sistema y el ámbito que lo rodea. Después se da una descripción en alto nivel de los componentes que forman el sistema, proporcionando una visión tanto estática como dinámica de los mismos. Por último se da una descripción más detallada de cada uno de estos componentes.

- El capítulo 4 describe la gestión del proyecto, mostrando cada una de sus fases a través de la planificación y el esfuerzo real dedicado.

- El capítulo 5 es el último y, como tal, presenta los resultados y las conclusiones finales del proyecto, desde un punto de vista tanto técnico como personal.

Después de estos cinco capítulos que resumen el proyecto, la memoria presenta una serie de anexos que contienen información adicional con el fin de completar el trabajo.

- El anexo A describe las tecnologías usadas durante el proyecto de forma detallada.

- El anexo B presenta el análisis de requisitos que se elaboró en un principio, pero que al ser muy extenso se resumió para el capítulo 2.

- El anexo C describe el resto de detalles del diseño que no aparecen en el capítulo 3.

- Por último, los anexos D y E presentan los manuales de usuario y de instalación, respectivamente.

Capítulo 2

Análisis del problema

En este capítulo se detalla el análisis del problema. La sección 2.1 presenta los principales objetivos que aborda este proyecto. La sección 2.2 enumera los requisitos concretos correspondientes a los objetivos previos.

2.1 Objetivos generales

El objetivo principal de este proyecto es facilitar la impresión de documentos, reportes o etiquetas, que puede generar la aplicación SILO de HP, mediante un driver unificado. Se trata de automatizar la generación del código adecuado para la impresión. Para ello son necesarias dos aplicaciones. Una aplicación *front-end* que permita al usuario la generación de las plantillas y la impresión de documentos, así como la configuración de algunas opciones de impresión de los dispositivos. La otra aplicación, el *motor*, debe atender las peticiones de impresión que provengan tanto de la aplicación *front-end*, como de otras aplicaciones. También debe gestionar las peticiones de generación de plantillas que reciba de la aplicación *front-end*. Uno de los propósitos de tener estas dos aplicaciones, aparte de la gestión de plantillas y la impresión de documentos, es que se puedan atender dos tipos de peticiones: atendidas y desatendidas. Las peticiones "atendidas" son todas aquellas en las que intervienen los usuarios y las "desatendidas" son las que provienen de otras aplicaciones y no necesitan obtener los datos de entrada del usuario. Para conseguir todo esto se han considerado los siguientes objetivos:

Objetivo 1

- Se necesita una herramienta que permita generar plantillas - *templates* - de los documentos que se van a imprimir. Una plantilla determina la estructura básica de un documento y contiene su configuración, almacenando la disposición y el formato de cada uno de los elementos que la componen. Cada plantilla se asociará con un tipo de documento: factura, albarán, etiqueta o informe. Estos tipos de documentos corresponden a los *outputs* estándar que genera la aplicación SILO. Cuando hablemos de *outputs* nos referiremos a los documentos finales que pretendemos obtener como resultado de su impresión. Además de definir sus elementos, es necesario conocer el dispositivo en el que se imprimirá. Por tanto, la configuración de las plantillas deberá incluir, además de los elementos que la componen, las dimensiones totales del área donde se van a posicionar y el dispositivo que se usará para su impresión. El usuario debe interaccionar con esta aplicación, que le guiará

paso a paso durante la creación de plantillas y el usuario deberá rellenar la información necesaria para ello. Una plantilla podrá ser modificada o incluso duplicada para generar una nueva a partir de la original y guardar ambas.

- El segundo objetivo de esta parte de la aplicación o *front-end* que permite la creación y edición de plantillas es proporcionar la posibilidad de imprimir documentos. Se denomina *front-end* porque es la parte visible de la aplicación que interacciona con los usuarios. Para la impresión de un documento se necesita, en primer lugar, seleccionar su plantilla correspondiente. La plantilla proporciona gran parte de la información necesaria para la impresión pero no toda, ya que a partir de una misma plantilla se pueden obtener distintos documentos. Esto es posible debido a que almacenará unos elementos con contenido fijo y otros con contenido variable. El usuario es el que proporciona esos datos variables cuando solicita la impresión de un nuevo documento. La plantilla tiene asociado un dispositivo para la impresión de los documentos que se generen a partir de ella. Se debe permitir al usuario modificar algunas de las opciones de impresión del dispositivo. En algunos dispositivos se permite la impresión de varios documentos utilizando una misma plantilla, por tanto también se soportará esta opción.

- Otro objetivo es ofrecer a determinados usuarios la posibilidad de configurar los dispositivos de impresión para evitar que cualquier usuario pueda cambiar algunas de las opciones de impresión.

Objetivo 2

- Los objetivos que se han presentado hasta el momento hacen referencia a la parte de la aplicación que es capaz de atender las peticiones de los usuarios de forma directa. El último de los objetivos consiste en poder imprimir documentos de forma desatendida, es decir, por orden de otra aplicación a través de un mensaje que se lo indique. Se necesita tener una aplicación principal, *motor*, que sea capaz de atender estos dos tipos de peticiones (atendidas y desatendidas) y que realmente se ocupe de almacenar las plantillas generadas y de traducir los documentos que se obtendrán como resultado final. Debe ser capaz de atender cualquier petición que reciba, sin importar el tipo. Además podrá gestionar más de una petición al mismo tiempo. Si el tipo de petición es "atendida" procederá de la parte de *front-end*, mencionada antes, y se le proporcionará información relativa a las plantillas o a los dispositivos y sus capacidades. En cambio, si la petición es "desatendida" no se necesita la intervención del usuario, ya que se deberá proporcionar toda la información necesaria para la impresión del documento solicitado. Dicha información la formarán la plantilla, los datos que la complementan, el dispositivo y sus opciones de impresión.

2.2 Análisis de requisitos

Esta sección contiene los requisitos definidos para nuestro sistema. Se detallan los principales aspectos y funcionalidades que presenta la aplicación. Se han agrupado todos aquellos que guardan cierta relación para acortar el número de requisitos elaborado en un principio, que puede verse en el anexo B. A continuación se enumeran todos los requisitos del sistema diferenciando entre requisitos funcionales y no funcionales.

2.2.1 Requisitos funcionales

Objetivo 1

R1: Todas las plantillas se corresponderán con uno de los tipos de documentos predefinidos y se podrán acompañar de una breve descripción para que el usuario pueda reconocerlas fácilmente.

R2: Las plantillas tendrán un tamaño determinado y se podrá delimitar su contenido definiendo unos márgenes. Podrán tener tantas páginas y elementos como sea posible disponer en ellas. No permitirá la superposición de elementos ni la colocación de los mismos fuera de los límites de página establecidos.

R3: Las plantillas se deberán asociar a uno de los dispositivos de impresión que estén disponibles. Estos dispositivos formarán parte del entorno de la aplicación SILO. El tamaño de plantilla se elegirá en función del tamaño de papel soportado por el dispositivo.

R4: La aplicación *front-end* solicitará un nombre y contraseña, de modo que cada usuario sólo tenga acceso a la información creada por él mismo. Así se realizará la gestión de las plantillas de cada usuario por separado. Se podrá tener un nombre de usuario común que represente a un grupo de usuarios que vayan a compartir las mismas plantillas para generar sus documentos.

R5: Cada usuario dispondrá de su propia lista de plantillas al identificarse. Las plantillas se representarán con un nombre único, elegido por el usuario en el momento de crearlas.

R6: La aplicación guiará al usuario paso por paso indicándole la información necesaria para la creación de plantillas. Los datos introducidos deberán ser completos y del tipo y forma especificados. Al terminar de crearla, se guardará la plantilla junto con su información correspondiente.

R7: Las plantillas creadas se podrán editar en cualquier momento con el fin de modificar su configuración o alguno de los elementos que formen parte de ellas. Se ofrecerá la posibilidad de modificar una plantillas ya creada o de duplicarla para generar una nueva a partir de la original y conservar ambas. Otra operación disponible será la eliminación de plantillas.

R8: Se dispondrá de una base de datos en la que se almacenará toda la información referente a los *inputs* o datos de entrada necesarios para rellenar la plantilla, el formato y configuración de las plantillas y los dispositivos y capacidades de los mismos.

R9: La generación de los documentos a imprimir requerirá la selección de una de las plantillas junto con unos datos de entrada que completen su contenido. Estos datos de entrada o *inputs*, se proporcionarán a través de un fichero de texto plano o con formato XML, en el caso de peticiones de usuario. Si las peticiones provienen de otras aplicaciones se obtendrán a partir de un mensaje con un formato predefinido.

R10: Se soportará la impresión de varios documentos utilizando una misma plantilla en aquellos dispositivos en los que sea posible.

R11: Se podrán seleccionar las opciones de impresión correspondientes al dispositivo asociado a la plantilla. Si no se selecciona ninguna opción, se imprimirán los documentos con la configuración actual del dispositivo.

R12: Algunos usuarios podrán configurar las opciones de impresión modificando los valores de fábrica establecidos. Aunque se guarden nuevas configuraciones se podrán recuperar estos valores iniciales.

R13: Se podrá probar la nueva configuración del dispositivo antes de ser guardada, imprimiendo una página de prueba para comprobar que sea correcta.

R14: La aplicación *front-end* usará pantallas emergentes para la comunicación de errores o mensajes de información al usuario.

R15: Las pantallas de la aplicación permitirán la identificación del usuario, ver o cambiar las opciones de configuración, ver las opciones de plantillas que permitan su creación, edición o eliminación, ver o cambiar algunas opciones de impresión y seleccionar un archivo con los datos que se van a incluir en la plantilla seleccionada para su envío o impresión.

Objetivo2

R1: La aplicación tendrá una parte o núcleo principal, el *motor*, capaz de atender todas las peticiones que se reciban, ya sean de usuarios o de otras aplicaciones ajenas a nuestro sistema. Deberá gestionar toda la información referente a usuarios, plantillas, documentos y dispositivos.

R2: Las peticiones de usuarios llegarán a través del *front-end* de la aplicación, que es la parte encargada de interaccionar con ellos. Su función principal será realizar el intercambio de información necesario entre el núcleo o motor principal de la aplicación y los usuarios para generar nuevas plantillas, modificarlas o imprimir documentos.

R3: Se generará un fichero XML por cada documento que se vaya a imprimir. Este fichero almacenará toda la información necesaria para su impresión siguiendo un esquema predefinido.

R4: Se examinarán las capacidades del dispositivo, comprobando que la impresión del documento sea factible. En caso de no ser posible, se realizará una búsqueda de equivalencias aprovechando al máximo sus capacidades con el fin de imprimir todos los elementos del documento de forma correcta.

R5: Si las capacidades del dispositivo no permiten la impresión correcta de todos los elementos del documento, éste se imprimirá igualmente.

R6: Se almacenarán también plantillas de estilo XSL necesarias para generar todos los tipos de outputs posibles en el código nativo del dispositivo. Habrá una hoja de estilo por cada lenguaje de descripción del dispositivo al que es capaz de traducir la aplicación. En total serán tres los lenguajes elegidos: PCL, ZPL y PGL.

R7: La transformación de los documentos al lenguaje nativo del dispositivo se producirá a través del fichero XML generado y la hoja de estilo XSL correspondiente al lenguaje.

R8: Se almacenarán los documentos listos para imprimir en la localización indicada al crear su plantilla y se dará por finalizada la petición de impresión.

R9: Antes de empezar a usar la aplicación el usuario deberá realizar la instalación de la misma mediante un archivo de "setup". Durante el proceso de instalación se pedirá al usuario la aceptación de las licencias de uso correspondientes y se seleccionará un directorio destino para almacenar los archivos necesarios que permitan su funcionamiento.

2.2.1 Requisitos no funcionales

R1: El sistema permitirá que se almacenen tantas plantillas e información correspondiente a los documentos, como sea posible en la base de datos, mientras no afecte a la eficiencia del sistema.

R2: La aplicación necesitará un motor de base de datos, existente en el producto SILO de HP. El sistema de gestión de la base de datos será Oracle, a partir de la versión 10.

R3: El sistema operativo sobre el que se desarrollará la aplicación será Linux Red Hat, en cualquiera de sus versiones 5 ó 6.

R4: La parte *front-end* de la aplicación requerirá tener instalada la máquina virtual Java a partir de la versión 6.1.3.

R5: El núcleo o motor de la aplicación se desarrollará en lenguaje C, utilizando las librerías que se consideren necesarias y las consultas a la base de datos se embeberán en el código C mediante el uso de PL/SQL.

R6: La aplicación podrá ser ejecutada en cualquier Sistema Operativo siempre y cuando contenga la máquina virtual de Java y tenga acceso a Internet.

R7: Resultará imprescindible adquirir los distintos módulos que componen el producto SILO, ya que se aprovecharán los recursos necesarios.

R8: La comunicación entre los clientes, que realizan las peticiones desde la parte de *front-end*, y el motor de la aplicación se realizará mediante sockets[12][13].

R9: Las peticiones que provengan de otras aplicaciones se comunicarán con el motor de la aplicación haciendo uso de la sincronización de procesos y la base de datos.

R10: Se utilizarán parsers XML para analizar los documentos con dicho formato.

R11: Se necesitarán motores de transformación XSL (XSLT) para convertir los documentos de entrada con formato XML en el formato de salida deseado.

R12: La aplicación dispondrá de un conjunto de pantallas en formato gráfico, cuyo idioma será el español. Los colores y secciones que conformarán la interfaz cumplirán los estándares de accesibilidad y usabilidad.

R13: La aplicación no presentará unas grandes capacidades de usabilidad, en cuanto aprendizaje y comprensión de uso del mismo por parte de un usuario no técnico, que no tenga ciertos conocimientos informáticos.

R14: La aplicación será robusta y tolerante ante errores previstos que puedan ocurrir, siendo capaz de recuperarse tras la aparición de alguno y evitando la pérdida de información. Lo hará con un costo mínimo, de tal manera que no modifique en exceso el rendimiento óptimo del sistema para continuar con su funcionalidad.

R15: La aplicación no ofrecerá la posibilidad de modificación o actualización de la base de datos, estas operaciones serán realizadas por un administrador externo al sistema.

R16: Se aplicará un proceso de ingeniería del software basado en una serie de métodos y técnicas que servirán de guía durante todo el desarrollo del proyecto, con el fin de mejorar su calidad.

Capítulo 3

Diseño de la solución

En este capítulo se describe el diseño adoptado desde una perspectiva global partiendo del entorno que lo rodea hasta llegar a detallar los elementos que lo componen. En la sección 3.1 se explica el contexto de ejecución de la aplicación y cómo interactúa con los elementos que la rodean, así como la arquitectura del sistema. La sección 3.2 describe el diseño en alto nivel de los componentes del sistema desde los puntos de vista estático y dinámico.

3.1 Arquitectura del sistema

El sistema desarrollado se creó con el fin de ser incluido dentro del producto SILO perteneciente a HP. Debido a esto, parte del diseño referente a la base de datos y la comunicación entre procesos han sido adaptados al desarrollado previamente por la empresa. La aplicación SILO se compone de una serie de módulos, junto con los que se integrarán los que se han creado para este proyecto. Al tratarse de un driver formará parte de los módulos que realizan las funciones básicas de la aplicación. SILO es una solución software que permite a una empresa de Logística y Transporte gestionar sus almacenes en tiempo real.

La figura 3.1 muestra el entorno de la aplicación SILO. Se pueden ver diferentes tipos de conexiones. Por un lado están los usuarios conectados a través de la Web de la aplicación SILO que ofrece acceso a todos sus servicios. Se pueden ver también las conexiones que realiza con distintos tipos de dispositivos, ya sean de impresión o de gestión de almacenes. Por último se incluyen las conexiones con la empresa HP, desarrolladora de la aplicación SILO, que realiza un mantenimiento constante sobre ella. La aplicación SILO está compuesta por una serie de módulos claramente diferenciados según sus funcionalidades. Algunos de estos módulos se comunican con la base de datos de SILO con el fin de obtener y almacenar la información necesaria. La Web o el software instalado en los dispositivos de gestión de almacén sirven como un medio para presentar la información al usuario y recoger todas sus peticiones.

La aplicación diseñada para este proyecto tiene claramente dos partes diferenciadas, que se pueden ver en la figura 3.1 como parte de la aplicación SILO en una caja con el nombre ”Aplicación de impresión”. La parte principal la compone el motor de la aplicación que actúa como un servidor y permanece activo a la espera de recibir peticiones de usuarios u otros procesos que forman parte de la aplicación SILO. Es un

sistema multitarea, por lo que puede atender varias peticiones a la vez a medida que le vayan llegando. El motor de la aplicación es el que gestiona toda la información necesaria para la impresión de los documentos. Se encarga de atender todas las peticiones que se reciben, ya sean conexiones de usuarios a través de la interfaz o mensajes de otros procesos que se almacenan en la base de datos. Además realiza todas las conexiones con la base de datos para obtener u extraer información acerca de plantillas, documentos o dispositivos. La otra parte del sistema la compone el *front-end* de la aplicación, que es una interfaz cuya función es facilitar al usuario la creación de plantillas y la impresión de documentos. De este modo se evita que las plantillas, se almacenen directamente en la base de datos o se definan en forma de ficheros. Este el principal motivo por el cual la aplicación ofrece una interfaz formada por una serie de pantallas, cuya finalidad es servir de soporte al usuario y proporcionarle ciertos servicios a realizar de manera más cómoda. Se debe destacar también la opción de configurar los dispositivos que proporciona a aquellos usuarios que tengan permiso para administrarlos.

Un ejemplo de uso de esta aplicación de impresión que se ha desarrollado podría ser la creación de una nueva etiqueta para una serie de *palets* de uno de los almacenes que gestiona la aplicación SILO. Al crear la etiqueta se le asignaría uno de los dispositivos de impresión que estén disponibles. A partir de ese momento se podrían imprimir tantas etiquetas como se desee en ese dispositivo.

Se ha diseñado de forma que sea posible incorporar tecnologías futuras, ya que los detalles de los dispositivos son abstraídos. En esencia, el driver universal funciona como un controlador de impresión tradicional desde la perspectiva final de los usuarios.

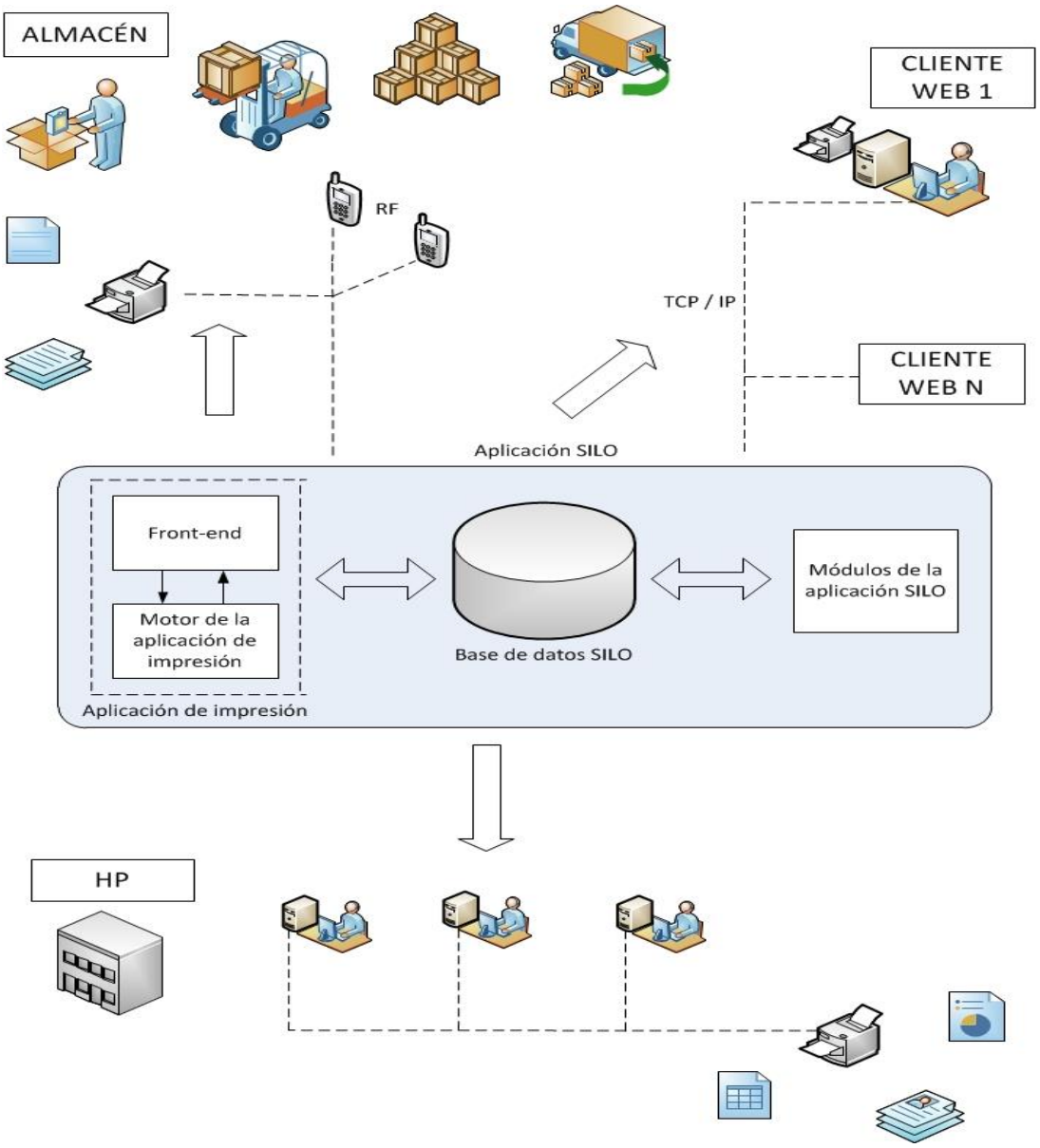

Figura 3.1 Entorno de la aplicación SILO

3.2 Descripción en alto nivel del sistema

3.2.1 Visión estática

La funcionalidad del sistema se definió en el documento de análisis de requisitos. En la figura 3.1 se ha visto el entorno de la aplicación SILO y en esta sección se va a describir la *aplicación de impresión* resaltada en la figura dentro de la aplicación SILO. La figura 3.2 muestra los componentes principales del sistema. En secciones anteriores se ha hablado de dos partes diferenciadas dentro de la aplicación, siendo la interfaz la que representa la parte de *front-end* y el resto corresponde al motor de la aplicación. Este motor lo componen: el *gestor de entradas*, el *gestor de la base de datos* y el *motor de*

impresión junto con el *motor de transformación*. En la figura se puede observar que la comunicación entre estas dos partes es por medio de *sockets*, por tanto toda la información que se transmitan será un mensaje en formato de bytes que tendrán que codificar y decodificar siguiendo un patrón establecido. A continuación se describen todos sus componentes destacando el papel que desempeñan y la relación que mantienen unos con otros.

La interfaz sirve para interaccionar con los usuarios y se ocupa de recoger y presentar la información relevante en cada momento. Se compone de una serie de ventanas que son las que muestran las operaciones disponibles a realizar sobre las plantillas y la impresión de documentos. Estas ventanas están formadas por una serie de elementos gráficos (campos de texto, listas de elementos, botones…etc) sobre los que el usuario puede realizar distintas acciones, que serán controladas por los gestores asociados a esas ventanas. En la figura 3.2 se diferencia claramente entre las *ventanas* y los *gestores* de éstas, que son clases creadas para realizar las operaciones necesarias sobre los datos que provengan del usuario o del motor de la aplicación. Todos los datos manejados hacen referencia a las plantillas del usuario, los elementos de las plantillas, los documentos, los dispositivos disponibles y sus capacidades.

El *front-end* y el *motor* de la aplicación de impresión intercambian una serie de flujos de entrada y salida a través de *sockets*. El tipo de información que se transmite en cada momento depende del tipo de petición realizada. Durante la creación de plantillas se solicitará al *motor* de la aplicación una lista de dispositivos disponibles y algunas de sus capacidades y al finalizar la creación se le enviará toda la información de la plantilla. Durante la impresión de documentos se necesitará una lista de plantillas disponibles para seleccionar una y un fichero de datos que la complete. El *motor* de la aplicación deberá proporcionar las plantillas, los dispositivos asociados a cada una y las opciones de impresión disponibles con sus valores por defecto. El usuario será el que proporcione uno o varios ficheros de datos para elaborar los documentos a imprimir. Todas estos datos necesarios para la impresión se enviarán al *motor* de la aplicación para que complete el proceso de impresión.

El *gestor de entradas* es el que controla todo lo que entra y sale del sistema, reconociendo el tipo de entrada y analizando su contenido. Por un lado, se comunica con la *interfaz* por medio de *sockets* enviando y recibiendo información, en bytes, y codificándola o decodificándola. Por otro lado, reconoce la llegada de señales de otros procesos que forman parte de SILO y las interpreta como un evento que le indica que debe consultar la base de datos a través del módulo que gestiona estas conexiones. Los datos recibidos por parte de la interfaz los analiza y decodifica la cabecera del mensaje, que es la que le indica la operación que debe realizar. Una vez que detecta el tipo de operación a realizar, sabrá si debe enviar la información que le solicitan o si debe analizar el resto de mensaje y almacenar su contenido en la base de datos. Utiliza un conjunto de estructuras de datos para organizar la información que recibe y que comparte con el resto de módulos. En caso de recibir información referente a la generación de plantillas se comunicará únicamente con el módulo que gestiona las

conexiones a la base de datos. En cambio, si se trata de una orden de impresión de documentos deberá comunicarse con el *motor de impresión*, aunque también requerirá los servicios del otro módulo.

El *gestor de la base de datos* se encarga de intercambiar información entre la base de datos y el resto de módulos del *motor* de la aplicación. Esta información la organizan en estructuras de datos conocidas por todos ellos. Su función principal es la comunicación con la base de datos para guardar la información necesaria en las tablas adecuadas u obtener aquella que se requiera para la impresión de documentos o para mostrar cierta información a los usuarios a través de la interfaz. El único que le proporciona información para ser almacenada en la base de datos es el *gestor de entradas*, ya que el *motor de impresión* sólo se comunica con él para extraer los datos que necesite. Las estructuras de datos que intercambia con el *gestor de entradas* se encuentran representadas por cajas en la figura 3.2 dentro de cada uno de los bloques de estos módulos. Estas estructuras contendrán todo lo referente a las plantillas, los elementos que contienen, los documentos a imprimir, los dispositivos y las opciones de impresión o configuración con sus valores actuales.

El *motor de impresión* es el encargado de gestionar las peticiones de impresión de documentos que recibe desde el *gestor de entradas*. Realiza todas las acciones necesarias para generar un fichero en formato XML, que contendrá toda la información relativa a los documentos que se van a imprimir. Este fichero es el que utilizará el *motor de transformación* para obtener el resultado final. La petición debe proporcionarle el nombre de la plantilla y usuario al que pertenece, los datos del documento que completan la plantilla y las opciones de impresión seleccionadas. La configuración y formato de la plantilla, así como la información relativa al dispositivo y sus capacidades deberá extraerla de la base de datos a través del módulo que gestiona sus conexiones. Además de recopilar todos estos datos en un fichero, otra de sus funciones principales es analizar las capacidades del dispositivo que va a realizar la impresión del documento. Deberá comprobar todos los elementos del documento para ver si es posible su impresión directa o si debe realizar algún cambio que permita su impresión aprovechando las capacidades del dispositivo. Cualquier cambio se verá reflejado en el fichero XML.

En la figura 3.2 se pueden ver los componentes internos del *motor de transformación*. El *gestor XML* sería el que realiza todas estas operaciones relacionadas con la creación y modificación del fichero XML y el otro componente, el *gestor de imágenes*, que se encarga de convertir los distintos formatos de imágenes a uno común que se pueda incluir en el fichero XML, lo complementa. El *gestor de capacidades* se encarga de verificar que el dispositivo puede realizar la impresión del documento, analizando todas sus capacidades para ver si son compatibles. En caso de que no lo fueran, trataría de encontrar una solución para realizar una impresión correcta. Los formatos de entrada admitidos para las imágenes que aparezcan en los documentos son *jpeg* y *bitmap*, por ser de los más utilizados. Los ficheros de tipo XML no admiten contenido binario y por

este motivo se deben transforman las imágenes a un formato que se pueda incluir, que es el de *64 bits*.

Como último componente dentro de este bloque se encuentra el *motor de transformación*, que es el encargado de obtener el resultado final. Recibirá como entrada el fichero XML con toda la información necesaria para la impresión y se le indicará el *parser* o traductor que debe emplear para obtener como resultado un fichero traducido al lenguaje nativo del dispositivo. Lo componen tres *parsers* cuya función es añadir información al fichero XML, que ayudará a realizar su traducción. Cada *parser* hace referencia a uno de los lenguajes de descripción de dispositivo admitidos: *PCL*, *ZPL* y *PGL*. Una vez añadida la información al fichero XML, realiza la transformación final a través de hojas de estilo XSL. Estas hojas XSL son ficheros que contienen código que sirve para transformar el contenido de ficheros XML y obtener el formato deseado. En este caso, se usan para convertir la información contenida en el fichero XML a una serie de comandos y parámetros correspondientes al lenguaje nativo del dispositivo.

Todos los componentes mencionados realizan un control de errores para evitar posibles fallos del sistema. Se ha tenido en cuenta que pueden producirse situaciones incorrectas que obligan a controlar de alguna manera estos eventos excepcionales. Esto es necesario, por ejemplo, para evitar accesos erróneos a la base de datos o errores de introducción de los datos a través de la interfaz. El *gestor de entradas* manejará estos posibles fallos, ya que es el encargado de comunicar con la *interfaz*, que en estos casos deberá mostrar un mensaje de error a través de una ventana para informar al usuario sobre lo ocurrido. Esto puede provocar en algunos casos el abandono del programa y en general, contribuirá a la robustez del sistema.

Se han explicado todos los componentes de la figura 3.2, pero como se puede ver, la aplicación también está formada por una *base de datos*, que se encuentra en el servidor Oracle y es la que HP diseñó para la aplicación SILO. La *base de datos* contiene mucha información útil para todos los módulos que componen la aplicación SILO, pero la mayoría de estos datos no resultan interesantes para nuestra aplicación. La mayor parte de las tablas a las que accede nuestra aplicación han sido creadas para guardar toda la información relativa a las plantillas, a sus elementos, a los documentos y a las peticiones que atiende. La información relativa a los dispositivos y a las peticiones que llegan de otras aplicaciones ya existía y simplemente se ha añadido algún dato adicional para completar dicha información. En el caso de los dispositivos se necesitaba un fichero XML que contuviera sus capacidades.

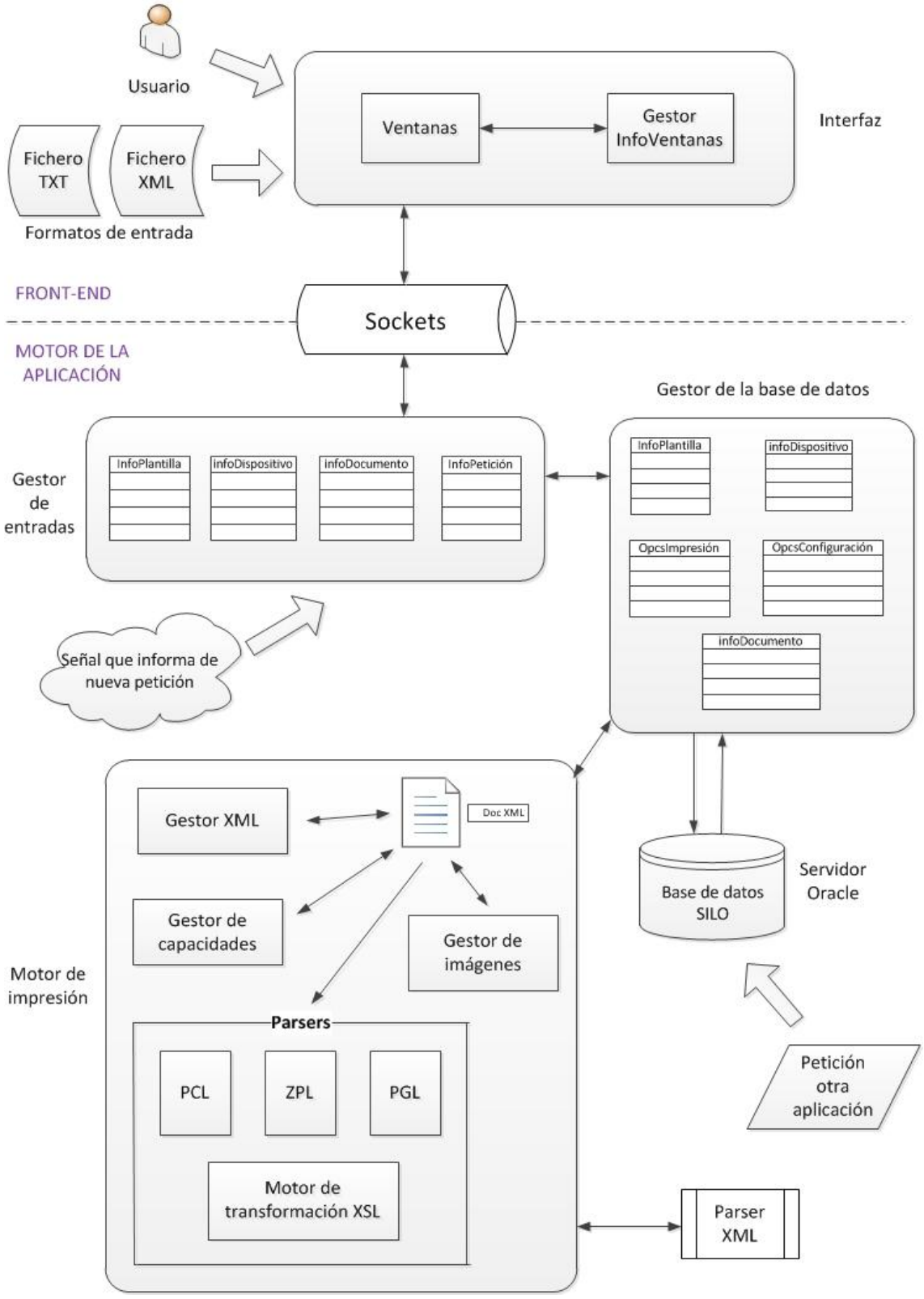

Figura 3.2 Visión estática de los bloques que componen la aplicación

3.2.2 Visión dinámica

El modelo dinámico muestra la forma en la que se comportan los componentes de nuestro sistema a lo largo del tiempo. Se presenta al sistema como una secuencia de eventos entre los elementos que lo componen. Para mostrar el modelo de comportamiento se indica la forma en que el software responderá a los eventos o estímulos externos.

Al examinar los casos de uso del sistema, se pueden identificar los eventos pertenecientes a cada escenario tomado por cada caso de uso. A continuación se exponen los diagramas de secuencia pertenecientes a tres casos de uso que representan los comportamientos fundamentales del sistema. En el primero se muestra la identificación de un usuario al iniciar la aplicación *front-end*. En el segundo el usuario realiza sus gestiones de forma atendida, introduciendo una serie de parámetros que se le indican para lograr la creación de una plantilla. El último caso corresponde a la orden de impresión de un documento a partir de una plantilla creada.

En la figura 3.3 se muestra la secuencia de eventos correspondiente a la identificación de los usuarios a través de la interfaz. Este caso de uso se produce siempre que el usuario utiliza la aplicación, ya que el inicio muestra una ventana que solicita un nombre de usuario y contraseña. En la figura se ve cómo el usuario introduce los datos y la *ventana* de inicio utilizará su *gestor de ventana* correspondiente para que guarde esos datos y los gestione. En este caso el *gestor de la ventana* recibe los datos del usuario para que envíe estos datos al *gestor de entradas*. El *gestor de entradas* debe aceptar la conexión y recibir los datos, identificando su cabecera para ver qué tipo de mensaje es y qué información contiene. Una vez que reconoce el tipo de operación que debe realizar, lleva a cabo la acción adecuada, siendo en este caso la comprobación del usuario. Se comunica con el *gestor de la base de datos* y le envía los datos del usuario para que los compruebe. Deberá notificarle al *gestor de entradas* si el usuario es correcto o no y en caso de ser correcto, el *gestor de la base de datos* deberá devolverle la lista de plantillas que posee ese usuario, junto con la descripción de esas plantillas y el dispositivo al que están asociadas. Una vez obtenidos todos los datos, el *gestor de entradas* los enviará por medio del *socket* que mantiene abierto mientras el *gestor de ventana* espera recibir dichos datos. Al recibir los datos, este gestor decodifica el mensaje recibido y si el usuario es correcto, guardará la lista de plantillas, sus descripciones y dispositivos y cerrará la conexión. En este caso se acepta al usuario y se muestra una *ventana* con el nombre del usuario y los datos acerca de sus plantillas. Si el usuario no fuera correcto se mostraría un mensaje de error y éste debería identificarse de nuevo.

El siguiente caso de uso representado en la figura 3.4 muestra la creación de una plantilla. La impresión del documento puede realizarse de forma opcional antes de salir de la aplicación. En este caso se ilustran las acciones principales que implican la creación de la plantilla, sin mostrar la impresión de un documento, ya que se mostrará en la figura siguiente. El punto de partida en este caso es la *ventana* correspondiente a la

pantalla de inicio que muestra la lista de plantillas y las distintas opciones que puede realizar el usuario, una vez que se ha identificado correctamente.

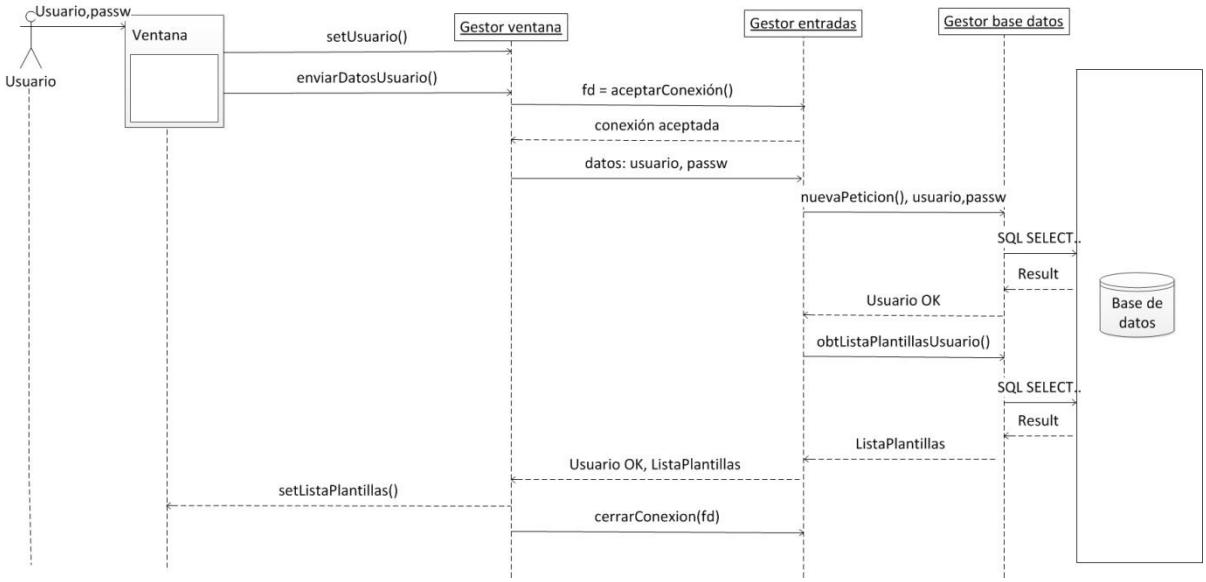

Figura 3.3 Diagrama de secuencia de la identificación de usuarios

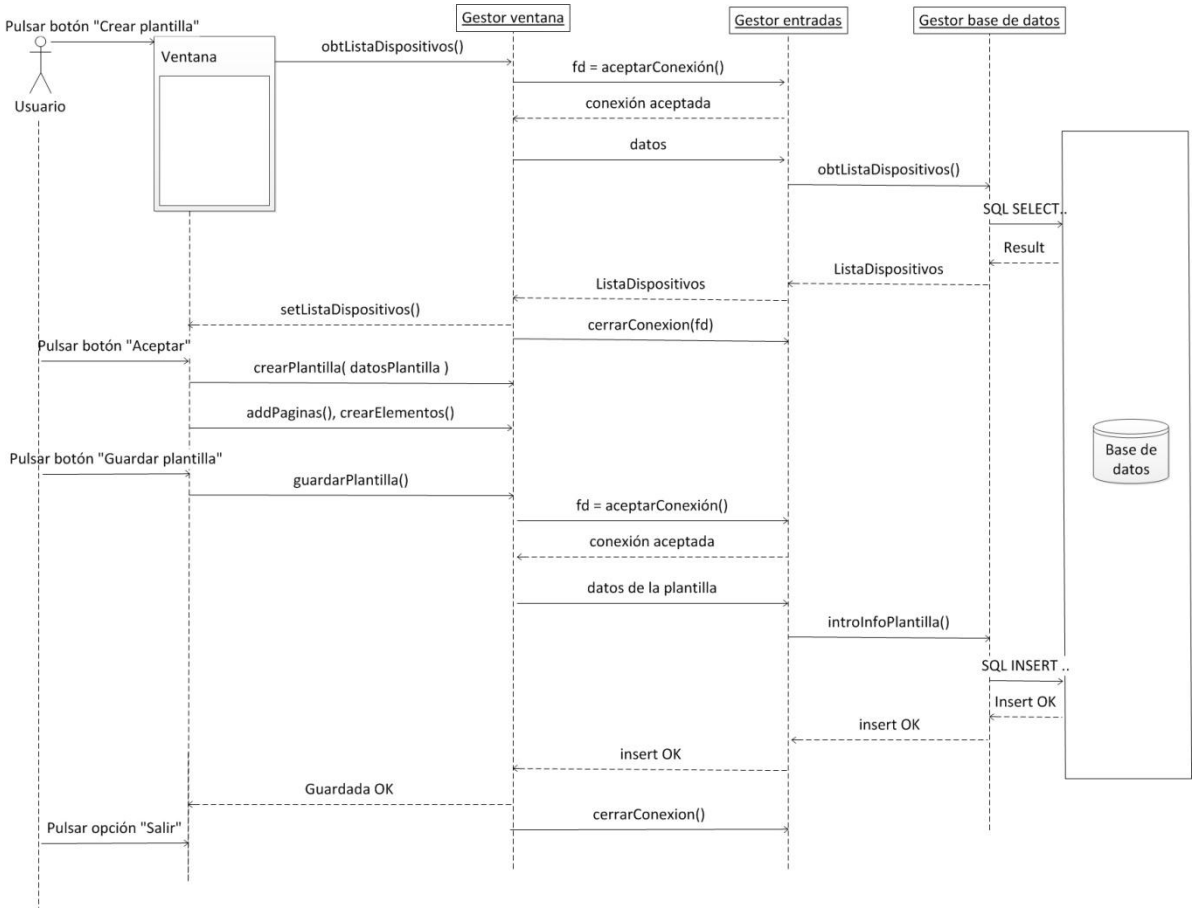

Figura 3.4 Diagrama de secuencia de la creación de plantillas

La figura 3.4 agrupa el conjunto de ventanas que van apareciendo en una sola que aparece como un único componente *ventana*. Como se aprecia en la figura, el usuario debe pulsar una serie de botones para avanzar hasta la ventana que muestre las primeras opciones para crear la plantilla. Antes de que aparezca esta ventana se usa su *gestor de ventana*, creado previamente, para obtener parte de la información a mostrar. Debe enviar un mensaje al *gestor de entradas* indicándole que debe obtener la lista de dispositivos disponibles en la base de datos. Además de enviar la lista de dispositivos, se incluirán también algunas de sus capacidades como el tipo de papel, su tamaño o resolución. Toda esta información es mandada por medio del socket y la recibe el *gestor de ventana*. Esta información se guarda y se muestra al usuario en la ventana que muestra los datos para crear la información general de la plantilla. El usuario debe rellenar todos los campos con la información necesaria como por ejemplo el nombre de la plantilla, su tamaño, tipo de documento, descripción, dispositivo,..etc. El *gestor de ventana* guarda estos datos y aparece una nueva *ventana* que va a permitir la creación de los elementos de la plantilla. El usuario puede ir añadiendo nuevas páginas y elementos a la plantilla. Toda la información relativa a las plantillas, a sus elementos, al dispositivo elegido y a sus capacidades la va guardando el *gestor de ventana* creado para ello. Además se encarga de comprobar las posiciones y dimensiones de todos los elementos para que no interfieran unos con otros y de esta forma permite o no su creación. Una vez que el usuario termina la creación de la plantilla pulsa el botón "Guardar plantilla", tal y como muestra la figura, y es el *gestor de ventana* el encargado de enviar toda la información de la plantilla al *gestor de entradas*, a través del *socket*. El *gestor de entradas* reconoce el tipo de operación a realizar y organiza la información de la plantilla en estructuras de datos para enviársela al *gestor de la base de datos* y que éste pueda almacenarla en las tablas disponibles para ello. Si la inserción ha sido correcta, se le comunicará al *gestor de entradas* y éste enviará un mensaje de confirmación al *gestor de ventana*, que al recibirlo cerrará la conexión. El usuario puede continuar con la impresión de un documento desde la siguiente *ventana* que se muestre o bien salir de la aplicación. Debido al exceso de operaciones a realizar, el caso de uso termina con el cierre de la aplicación y se continúa su ejecución en el siguiente caso de uso.

La figura 3.5 muestra la secuencia de eventos correspondiente al caso de uso de impresión de documentos utilizando la plantilla creada y comienza desde la última *ventana* del caso anterior. Para que el esquema de la figura 3.5 no resulte demasiado extenso se han agrupado todas las ventanas en una sola con el nombre de *ventana* y también algunas funciones. El usuario va solicitando o enviando parámetros de impresión, al igual que se hacía en el caso anterior. Cuando el usuario pulsa el botón "Opciones de impresión" se crea la nueva *ventana* que permitirá rellenar los datos necesarios para la impresión del documento. Al mismo tiempo, el *gestor de ventana* será el encargado de obtener cierta información del dispositivo y sus capacidades. Al igual que en los casos anteriores se necesita enviar un mensaje al *gestor de entradas* y éste gestione las operaciones a realizar. El *gestor de entradas* solicita la información al gestor de la base de datos, proporcionándole el nombre del dispositivo, y éste le

devuelve su estado, ubicación, lenguajes nativos de impresión y un conjunto de listas que contienen una serie valores correspondientes a sus capacidades, así como el valor seleccionado por defecto para cada una de ellas. El *gestor de entradas* codifica toda esta información y se la envía al *gestor de ventana*, siendo decodificada y gestionada para que pueda ser mostrada a través de la *ventana* adecuada. En esta *ventana* se podrán ver los datos generales del dispositivo y se tendrá acceso a una nueva *ventana* donde aparezcan las opciones del dispositivo referentes a sus capacidades. El usuario puede modificar los valores por defecto de las opciones de impresión que se muestren y guardarlas. La acción de ver la pantalla con las opciones de impresión es opcional. Si el usuario no solicitara ver dichas opciones se tomarían los valores por defecto contenidos en el fichero XML del dispositivo. El *gestor de ventana* es el encargado de almacenar el valor de estas opciones y una vez que termine, volverá a la *ventana* anterior. En esta *ventana* el usuario debe rellenar todos los datos necesarios para la impresión del documento, así como proporcionar un fichero de texto plano o de tipo XML para completar los datos de la plantilla seleccionada. Cuando el usuario termina pulsa el botón "Imprimir" y el *gestor de ventana* recopila toda esta información y le envía los datos al *gestor de entradas*. Éste decodifica la información y la reconoce como una petición de impresión. Este tipo de peticiones las gestiona el *motor de impresión*, pero antes el *gestor de la base de datos* almacena la información sobre el documento a imprimir: el nombre del documento y el contenido de los datos que completan la plantilla seleccionada. El motor de impresión solicita al gestor de base de datos todo lo que necesita para generar el fichero XML con la información de impresión del documento. Primero se obtiene toda la información relativa a la plantilla y al documento y se añade al fichero XML. Aunque en la figura no aparezca, si alguno de los datos es de tipo imagen el *gestor de imágenes* debe transformarlas al formato adecuado. A continuación se añaden también las opciones de impresión seleccionadas y por último se examinan las capacidades del dispositivo para ver si la impresión del documento es factible. En caso de no serlo, se buscarían alternativas para imprimir el documento con las capacidades del dispositivo disponibles. Al completar la creación del fichero XML, se realiza su transformación al lenguaje nativo del dispositivo mediante el *parser* y la hoja de estilo XSL correspondientes a dicho lenguaje. Al terminar todo el proceso, si el documento se obtiene correctamente se comunica al usuario el éxito de la operación a través de un mensaje informativo en una ventana emergente y se le muestra la opción de salir de la aplicación para terminar su ejecución.

La impresión de documentos de forma desatendida sigue las mismas acciones que en este caso, pero sin la intervención del usuario y obteniendo las peticiones a través de los telegramas que se almacenan en la base de datos.

En estas explicaciones sobre los casos de uso mostrados se habla de *ventanas* y *gestores de ventana* para no entrar en detalles. Si se hubieran mostrado en los diagramas todas las ventanas que intervienen, identificándolas por sus nombres, y todos sus *gestores de ventana*, que hacen referencia a objetos concretos que se crean en la aplicación *front-end*, hubieran sido demasiado extensos, poco legibles y difíciles de comprender. El

anexo C incluye unos diagramas que muestran la secuencia de operaciones con un mayor detalle.

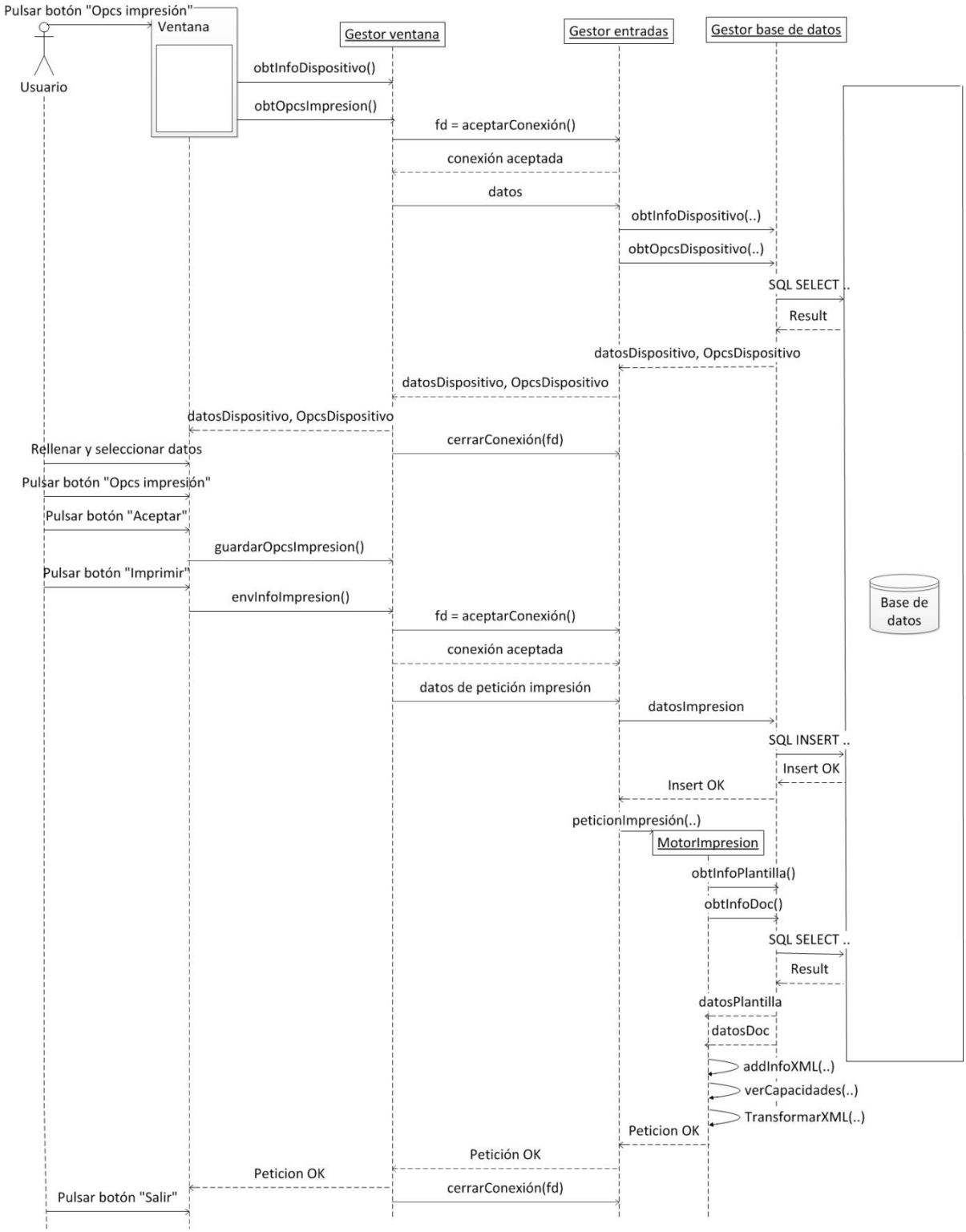

Figura 3.5 Diagrama de secuencia de la impresión de documentos

3.3 Diseño de los componentes del sistema

En esta sección se describe un diseño más concreto de los componentes del sistema. Se parte del diseño de la base de datos y se sigue con los correspondientes a los módulos principales del sistema: *gestor de la base de datos*, *motor de impresión* y *gestor de entradas* hasta finalizar con el diseño de la *interfaz*.

3.3.1 Diseño de la base de datos

La creación de la base de datos ha sido un punto clave del sistema, ya que toda la información acerca de las plantillas y dispositivos está almacenada en ella. El primer paso fue la construcción conceptual de un modelo de base de datos a partir del modelo existente de la base de datos que utiliza la aplicación SILO de HP y de los requisitos del sistema.

Esquema entidad-relación

En la base de datos se almacena todo lo referente a la información persistente del sistema. La figura 3.5 muestra un esquema de entidad-relación con toda esta información. Todas las entidades representadas en la figura corresponden a uno de los tres tipos de información relevantes en nuestro sistema, que son: la configuración de las plantillas y de todos sus elementos, los dispositivos y las peticiones de impresión de documentos que estén siendo atendidas por la aplicación. Según la figura 3.5, la información relativa a las plantillas la componen la entidad *Plantilla*, que almacena sus propiedades generales y todas las entidades que comienzan por *Elem_*, que hacen referencia a los distintos tipos de elementos que soporta la aplicación. Las entidades *Propiedades_elem* y *Dato_texto* son entidades débiles que contienen información relativa a los elementos de tipo texto. Los dispositivos sólo se representan en una única entidad *Dispositivo*, que es la que contiene toda la información relativa a sus propiedades y capacidades. Por último, las peticiones de impresión se representan con tres entidades: *Petición*, *Info_doc* y *Telegrama*. La entidad *Petición* representa a todas las peticiones que está atendiendo el sistema en ese momento, guardando cierta información sobre su estado, tipo y orden de llegada al sistema. La entidad *Info_doc* representa la información relativa a los documentos que se van a generar almacenando los datos de entrada que completan los elementos variables de la plantilla seleccionada. La entidad *Telegrama* hace referencia a las peticiones de impresión que llegan de otras aplicaciones y que se almacenan en la base de datos esperando a ser atendidas. Cuando son atendidas estas peticiones, se extrae su información, que hace referencia a la que representan las entidades *Info_doc* y *Petición*. Toda la información relativa a plantillas y dispositivos permanecerá en la base de datos, pero la relativa a las peticiones se borrará una vez que hayan sido atendidas.

Al tener una base de datos ya existente, con información acerca de los dispositivos y los usuarios de la aplicación SILO, se tuvo que adaptar el diseño conceptual al modelo de

datos establecido por HP. La base de datos de SILO contenía ya una tabla que almacenaba cierta información sobre los dispositivos a los que tenía acceso. Todos los dispositivos se identifican con un código numérico único y se almacena información sobre su nombre, tipo, estado, localización y otras propiedades sobre conexiones, protocolos o puertos que no son de interés para nuestra aplicación. El nombre, estado y localización se utilizan para proporcionar cierta información al usuario sobre el dispositivo en el que se va a imprimir. El tipo sirve para identificar los dispositivos de impresión, que son representados con el número 3. La información sobre las capacidades del dispositivo se ha añadido a la tabla *DISPOSITIVO* existente como un nuevo campo que pudiera contener los ficheros XML diseñados para describir los dispositivos de impresión utilizados. Debido a que no es una base de datos que soporte de forma nativa XML, se trata de un campo de tipo *clob*, que permite almacenar gran cantidad de información hasta un máximo de 4GB. Estos ficheros XML contienen principalmente características generales del dispositivo como el fabricante, modelo y lenguajes de impresión que soporta, así como todas las capacidades que posee en forma de lista de opciones de impresión con todos sus valores posibles y aquellos seleccionados por defecto. Cada capacidad se identifica con un número de opción, aunque también contiene su nombre para saber de qué opción se trata.

Otra información clave para adaptar nuestro sistema es la que se encuentra almacenada en las tablas que permiten la comunicación y sincronización entre los distintos procesos que componen la aplicación SILO. Estas tablas son las que almacenan los *Telegramas*, representados como entidad en la figura 3.5, que en el caso de nuestra aplicación hacen referencia a las peticiones de impresión de otras aplicaciones. Los formatos que deben seguir estos *Telegramas* son consultados por los procesos que forman la aplicación SILO para obtener la forma en la que deben enviar la información para que pueda ser comprendida por otro proceso. Los *Telegramas* deben contener la información necesaria para generar la petición de impresión: el nombre de la plantilla, el nombre del usuario, el nombre del dispositivo, el nombre del documento, los datos de entrada que complementan la plantilla y las opciones de impresión. Todos estos datos se organizarán siguiendo un formato preestablecido que se almacenará en la tabla *TIPO_FORMATO_TEL*. Esta tabla contiene todos los tipos de *Telegramas* que existen en la aplicación SILO, identificados con un número único, junto con su formato y su descripción. A las plantillas cuando son generadas se les asigna un nuevo tipo de *Telegrama*, que habrá que crear y almacenar en la tabla *TIPO_FORMATO_TEL*. La tabla *TL_IMPRESION* es donde se almacenarán las peticiones de impresión que se vayan recibiendo en forma de *Telegramas*. Esta tabla tiene cinco campos donde guardar el tipo de telegrama recibido, la fecha y hora de llegada, la prioridad, la secuencia y los parámetros con la información de la petición. Al igual que esta tabla, existen otras en la base de datos de SILO que almacenan otro tipo de telegramas, pero todas ellas poseen la misma estructura con los mismos campos.

La información relativa a las plantillas se almacena en la tabla *T_PLANTILLA*, que contiene su información general y como clave un número único que sirve para

identificarlas. Esta tabla contiene su nombre, el del usuario que la crea, el nombre del dispositivo, el tamaño de página, su orientación y sus medidas, el número de páginas y de elementos totales que contiene. No se permite asignar el mismo nombre a distintas plantillas pertenecientes a un mismo usuario, por tanto, también se pueden identificar mediante su nombre y el del usuario que las creó. Las plantillas se componen de uno o varios elementos distribuidos en una o varias páginas. Los elementos dependen de la existencia de las plantillas y por ello se han definido como una entidad débil en la figura 3.5. La base de datos contiene una tabla por cada tipo de elemento que puede contener una plantilla. Todos los elementos tienen unas propiedades comunes sean del tipo que sean. Estas propiedades se almacenan en la tabla *ELEM_PLANTILLA* y sirven para definir la posición del elemento en la plantilla, así como sus dimensiones. Todas estas posiciones y medidas se guardan en milímetros, ya que es una unidad de medida internacional que resulta fácil de convertir a otras unidades que se usan con frecuencia, como las pulgadas. Cada tipo de elemento posee sus propias características. Tras ver los distintos *outputs* o documentos que se suelen generar en la aplicación SILO, se seleccionaron los siguientes tipos para la creación de las plantillas:

- Texto: se refiere a cualquier palabra o conjunto de palabras que pueden aparecer en una línea en un documento. Cada dato de texto debe especificar su tipo: fijo o variable. Si es fijo tendrá asignado un texto que será su contenido y en caso de ser variable, se deberá definir el tipo de dato que contendrá (cadena, entero, real,…) y su longitud en número de caracteres. Este tipo de datos llevan asociadas unas propiedades referentes al tipo y tamaño de fuente, estilo, densidad,..etc. Se han agrupado como atributos de la entidad débil *Propiedades_elem*, que depende del elemento de texto al que hace referencia y cuya información se almacena en la tabla *PROPIEDADES_ELEM*. Los elementos de texto pueden ser fijos o variables. El texto fijo es aquel que aparece siempre en la plantilla y en el documento final tal y como se creó. El texto variable es el que varía de un documento a otro, tiene un nombre que lo identifica y una longitud predefinida. Todos los datos relativos a los elementos de texto se almacenan en las tablas *ELEM_TEXTO* y *DATO_TEXTO*. Este tipo de elementos se pueden agrupar por líneas cuando son creados, aunque en la base de datos se almacenará cada dato de texto por separado indicando su número de línea y su posición dentro de la misma. Por este motivo se emplean dos tablas, donde *ELEM_TEXTO* hará referencia al conjunto de los datos de texto y *DATO_TEXTO* almacenará las propiedades de cada dato.

- Imagen: se refiere a cualquier imagen que pueda ser incluida en un documento. Los formatos aceptados por la aplicación son *bitmap* y *jpeg* por ser los más comunes. Se caracterizan por un nombre que las identifique y el número de bytes que contengan por cada línea, así como el número de bytes en total. Los dispositivos de impresión manejan diferentes formatos de imagen y requerirán las conversiones de formato oportunas. Su información se almacena en la tabla *ELEM_IMAGEN*.

- Línea: se refiere a cualquier línea vertical u horizontal que pueda aparecer en un documento. Serán definidas a partir de su orientación, longitud y grosor.

- Caja: se refiere a un conjunto de cuatro líneas formando un rectángulo. Puede contener o no otros elementos en su interior. Se definirán a partir de la longitud de sus líneas verticales y horizontales, así como su grosor. Tanto este elemento como el de tipo línea almacenan sus propiedades en la tabla *ELEM_BORDE*.

- Código de barras: son elementos que suelen aparecer en las etiquetas y están soportados por las etiquetadoras. Sus propiedades varían entre los distintos códigos de barras que existen. Estos elementos almacenan su información en la tabla *ELEM_CODBARRAS*.

Se consideró añadir el elemento tabla, pero se optó por no hacerlo debido a la complejidad del elemento y a que puede formarse mediante la unión de varias líneas y de elementos de texto.

Las peticiones de impresión, que llegan tanto de los usuarios por medio de la interfaz como de otras aplicaciones por medio de la base de datos, almacenan su información en la tabla *T_PETICIONES* una vez que son atendidas. Aquí también se incluyen las peticiones de generación de plantillas por parte de los usuarios. La información que guarda esta tabla hace referencia al número de petición, que es único, al tipo (atendida o desatendida), al estado, al usuario, al nombre de plantilla y al nombre de documento a generar en caso de que sea una petición de impresión. En caso de que falle el sistema en algún momento, esto permitirá recuperar la información de la petición y el estado en el que se encontraba. Una vez que las peticiones se completen, serán eliminadas de la base de datos. La entidad *Petición* de la figura 3.5 hace referencia a toda esta información. La entidad *Info_doc* contiene atributos referentes al tipo de documento a imprimir: el nombre del documento, el número de copias a imprimir y el contenido de los elementos de texto de tipo variable junto con el orden que ocupan dentro de una página. El orden de los elementos se define empezando desde la esquina superior izquierda de las páginas y desplazándose de izquierda a derecha y de arriba a abajo. Todos estos datos se almacenan en una tabla llamada *INFO_DOC*. Esta tabla se rellena cada vez que llega una nueva petición de impresión con unos datos de entrada correspondientes a uno o varios documentos a imprimir.

A continuación la figura 3.5 presenta el esquema con todas las entidades que intervienen y sus relaciones. Algunas entidades presentan gran cantidad de atributos y por ello sólo aparecen algunos de los más importantes para que el esquema sea más legible.

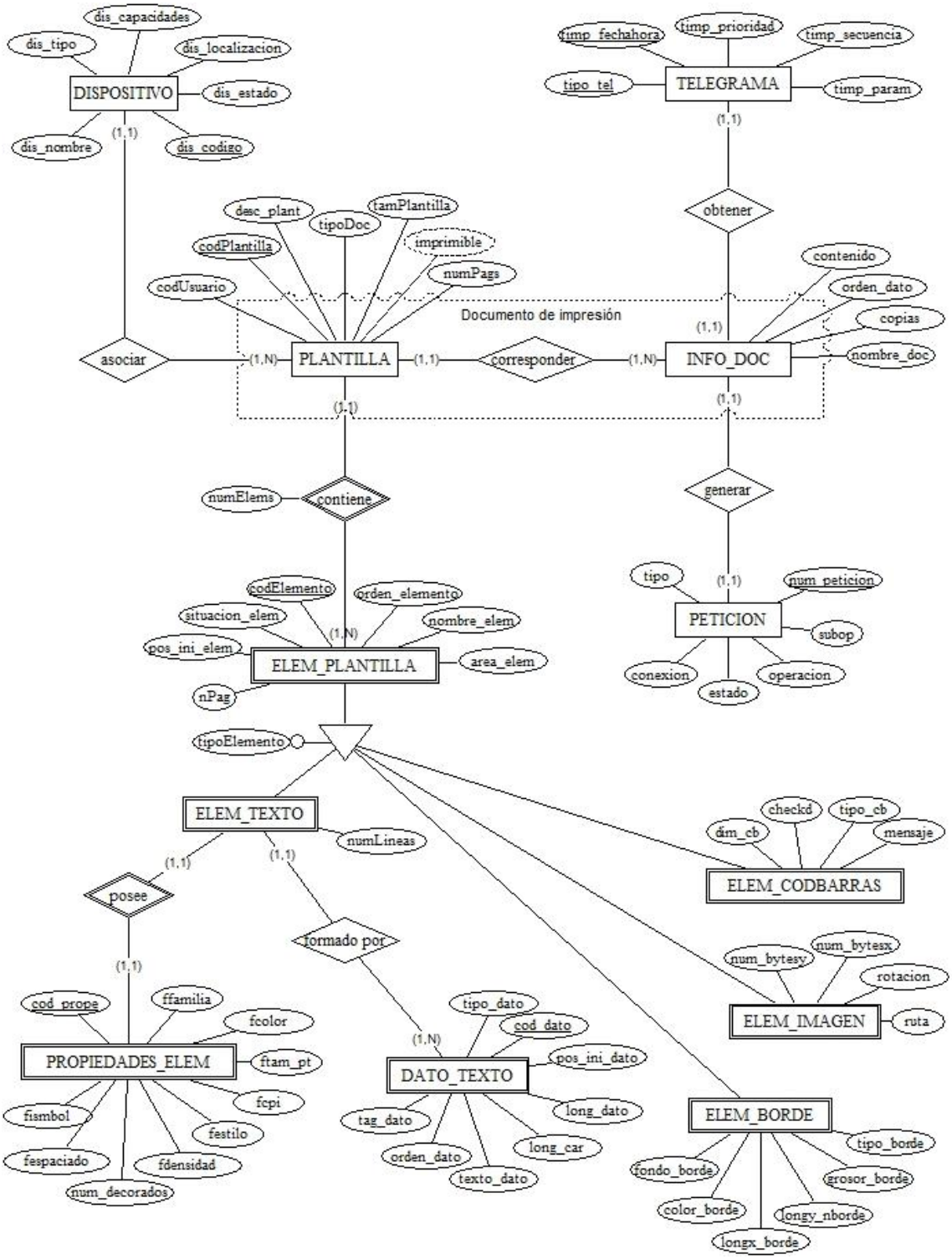

Figura 3.6 Esquema de entidad-relación de la base de datos del sistema.

3.3.2 Diseño del gestor de la base de datos

En esta sección se describe el diseño del módulo que realiza las conexiones con la base de datos. Su funcionalidad principal es consultar la información almacenada en BBDD.

Para ello también debe permitir guardar la información necesaria para poder ser consultada. Por tener estas dos funcionalidades claramente separadas del resto de la aplicación, se ha decidido agrupar todas las funciones necesarias en el módulo *gestor de base de datos*. Al relacionarse directamente con la base de datos, crea una capa de abstracción para el resto de módulos, siendo el único que maneja todas las tablas pertenecientes a la base de datos, junto con sus atributos. Se comunica directamente con el *motor de impresión* y el *gestor de entradas*, que se explicarán en secciones posteriores.

La información que debe almacenar la recibe del *gestor de entradas* en forma de estructuras o listas, compartiendo la especificación de las mismas. Tanto el *gestor de entradas* como el *motor de impresión* consultan la información de la base de datos a través de este módulo. La comunicación entre ellos se realiza mediante funciones que comprenden estructuras de datos comunes. Lo mismo ocurre con la inserción de la información. Estas estructuras comunes se visualizaban en la figura 3.2 y guardan la información relativa a las plantillas y sus elementos, a las peticiones y a los documentos junto con sus datos de entrada y opciones de impresión. Toda esta información que se agrupa en estructuras se corresponde con la que contienen los campos de las tablas mencionadas en la sección anterior.

Internamente funciona por un lado, obteniendo los datos de las estructuras o listas que recibe y realizando las consultas SQL de inserción de dichos datos en las tablas correspondientes. Por otro lado, cuando se trata de consultar la información realiza el proceso inverso haciendo las consultas SQL necesarias para obtener los datos requeridos y organizarlos en las estructuras de datos correspondientes. Por tanto se trata de un componente que recibe o crea las estructuras de datos que contienen la información que maneja la aplicación y realiza las consultas SQL adecuadas.

Permite guardar y extraer la siguiente información necesaria para el resto de módulos:

- Datos relacionados con la entidad *Plantilla*: proporciona información sobre todo lo referente a las plantillas, ya sea su información general o concreta acerca de cada uno de los elementos que la componen. Todos los datos extraídos se agrupan en una estructura llamada *infoPlantilla*, que contiene información general como el nombre de la plantilla, su descripción, el usuario que la ha creado, su tamaño, dispositivo asociado,…etc. Para los elementos de la plantilla se utilizan varias estructuras. La estructura *info_elem* agrupa una serie de datos comunes referentes a la posición y al tipo de dato. Las estructuras *elem_texto, elem_imagen, elem_codbarras, elem_borde* almacenan las características concretas de cada elemento, según el tipo de dato que su propio nombre indica. Además los elementos de tipo texto necesitan de una estructura adicional de tipo *prop_elem* para organizar las propiedades del elemento que hacen referencia a las características de la fuente utilizada.

- Datos relacionados con la entidad *Dispositivo*: proporciona su nombre, estado, localización o sus capacidades. Sólo se consultarán ciertos atributos que sean de

interés para nuestra aplicación, ya que la tabla *DISPOSITIVO* que almacena la información acerca de los dispositivos es utilizada por otras aplicaciones. Para este tipo de información no se utiliza ninguna estructura de datos, sino listas, que se generan para cada opción de impresión y sus valores correspondientes, y el fichero XML que contenga todas sus capacidades.

- Datos relacionados con la entidad *Petición*: proporciona información acerca del usuario que ha solicitado un servicio a nuestra aplicación, la conexión que está utilizando y el estado en que se encuentra la petición en ese momento. Según el estado, que hace referencia al tipo de operación que la aplicación está realizando, se puede obtener el documento a imprimir, si está llevando a cabo dicha acción, o la plantilla en la que se base un documento o que se esté creando por ejemplo. La estructura *status_silo* será la que contendrá todos estos datos.

- Datos relacionados con la entidad *Info_doc*: proporcionan información acerca de los elementos del documento a imprimir. Al igual que para las plantillas, se ha agrupado su información general en la estructura *info_doc*, que contiene el nombre del documento, de la plantilla, del usuario, del dispositivo, el número de petición, el número de copias,..etc. En lo que se refiere a las opciones de impresión seleccionadas, sólo se necesita el número de opción, correspondiente a una de las capacidades del dispositivo almacenada en su fichero XML de capacidades, y el valor elegido. Para la gestión de estas opciones se usa una lista. Para las opciones en las que no se seleccione ningún valor, se usará su valor por defecto. Por último se utiliza una lista para gestionar los elementos de texto variables seleccionados para rellenar la plantilla y convertirla en documento imprimible.

Una vez descritos todos estos componentes y las estructuras empleadas para guardar su información, se puede comenzar con el siguiente módulo.

3.3.3 Diseño del motor de impresión

En esta sección se describe el diseño del módulo que conecta con el resto de módulos. Recibe el nombre de *motor de impresión* porque sobre él recae la función principal del sistema: la impresión de los documentos. Se comunica con el *gestor de entradas*, que es el que le envía las peticiones de impresión para que las atienda y con el *gestor de la base de datos* para consultar la información que necesite. Atiende las peticiones y genera el fichero XML con toda la información necesaria para la impresión de documentos. Por último realizará la transformación del fichero junto con la hoja de estilo XSL adecuada para la obtención del fichero final traducido al lenguaje nativo del dispositivo de impresión.

En la figura 3.2 se podían ver varios subcomponentes que tienen sus funciones especificas dentro de este módulo. En primer lugar, tiene un *gestor XML*, que diferencia entre las peticiones que recibe desde la interfaz y las que llegan por medio de la base de

datos, realizando las acciones adecuadas para cada caso. Aunque la petición provenga de una fuente distinta, la información que se debe obtener es la misma: la configuración de la plantilla, los datos relativos al documento y a las opciones de impresión del dispositivo. Su función es recopilar esta información y organizarla en estructuras de datos similares a las que compartían tanto el *gestor de la base de datos* como el *gestor de entradas*. Estas estructuras contienen casi la misma información, pero reducida ya que sólo se requiere la información necesaria para añadirla al fichero XML que va a generar el *gestor XML*. El resto de acciones que debe realizar para completar la impresión del documento no difiere según el tipo de petición y las realizan el resto de componentes.

El gestor XML obtiene la información de las estructuras creadas y las va incluyendo en el fichero XML que debe generar haciendo uso de las funciones que emplean los *parsers* XML y que se encuentran en las librerías utilizadas para XML: *libxml*, *libxml2* [8] y *libxslt* [9]. La estructura que sigue este fichero se muestra en el anexo C a través de su esquema XSD correspondiente. Los ficheros XML se organizan mediante una serie de *tags* o etiquetas que sirven para reconocer el tipo de información que almacenan. Esta información hará referencia al nombre del documento, de la plantilla, del usuario, del dispositivo, así como a todas sus características relevantes. Primero aparecerán todos los datos relativos a la configuración de la plantilla como su tamaño, orientación, márgenes, dispositivo y resolución. A continuación se incluirán las opciones de impresión seleccionadas. Por último, se añaden todos los elementos que componen la plantilla organizados por páginas y enumerados por orden de aparición en el documento. Si se incluyen elementos de tipo imagen en el fichero XML, el *gestor de imágenes* deberá convertirlas al formato adecuado. Este componente maneja estructuras correspondientes a la información que se almacena en las cabeceras de ficheros con formato *bitmap* y *jpeg* y junto con los datos binarios propios de la imagen los codifica a formato de *64 bits*, que es el único que soporta un fichero XML, al estar basado en caracteres y no en binario.

Al terminar de recopilar la información a imprimir en el fichero XML, se deben analizar las capacidades del dispositivo asignado. Esta es la función que desempeña el *gestor de capacidades*. Ante la imposibilidad de imprimir alguno de los elementos del documento, se examinan las características disponibles en el dispositivo con el fin de aprovecharlas y buscar una solución para la impresión de esos elementos. Un ejemplo de esto son impresoras que no pueden imprimir códigos de barras y cuya función se puede suplir con la impresión de una imagen *bitmap* si son capaces de imprimir imágenes. Los elementos incompatibles con el dispositivo se sustituyen por aquellos que lo sean, modificando el fichero XML generado, y se da por terminado el análisis.

Una vez generado el fichero XML, dependiendo del dispositivo destino al que va a ser enviado, se traduce al lenguaje nativo apropiado invocando uno de los *parsers* disponibles: *PCL*, *ZPL* o *PGL*. Su función es analizar el fichero XML creado y añadir nuevos *tags* donde sea necesario incluir valores que usarán directamente los comandos del dispositivo. Por tanto, estos componentes realizan una primera traducción de los

datos que lo requieran, que son la mayoría. Se traducirán las características del papel a usar, la resolución, la simbología, las fuentes, las posiciones, los valores de las opciones de impresión,…etc. Las posiciones de los elementos utilizan como unidad de medida el milímetro, que en caso de no ser compatible con las que soporta el dispositivo, se transformarán para adaptarlas a la unidad de medida adecuada. Cuando se termine de analizar el fichero y añadir la información necesaria, se realiza la transformación final del fichero XML utilizando la hoja de estilo XSL adecuada, según el *parser*, y se obtiene el fichero de salida traducido al lenguaje nativo del dispositivo. La transformación se realiza mediante un motor XSLT, utilizando la librería *libxslt*. La hoja XSL aporta todos los comandos del lenguaje de impresión y sus parámetros los obtiene del fichero XML. Se deposita el fichero en la ruta indicada al crear la plantilla y se comunica la finalización con éxito de la petición. La aplicación no se encarga de gestionar la manera en la que el fichero llega a las colas de impresión, ya que otra parte de la aplicación SILO se encarga de ello.

3.3.4 Diseño del gestor de entradas

En esta sección se describe el diseño del módulo que recibe las peticiones de entrada de la aplicación, tanto de impresión como de generación de plantillas. Se comunica con la *interfaz*, realizando las conexiones a través de *sockets* y con el *gestor de la base de datos* y el *motor de impresión* mediante las estructuras de datos que comparte con ellos.

Debido a la diversidad de tipos de petición y de las fuentes de las que pueden provenir es un componente capaz de recibir y atender concurrentemente cualquier petición de entrada. Las peticiones realizadas por usuarios a través de la *interfaz* envían y reciben una serie de datos por medio del *socket* que los conecta. Una misma petición que proceda de la *interfaz* puede realizar varias conexiones a través del *socket* hasta que finalice con éxito. Es por esto que en la primera conexión que se realiza desde la *interfaz*, cuando el usuario entra en la aplicación, es necesario asignar un número único a la petición para reconocerla en conexiones posteriores y guardar su estado en la base de datos. Cuando atiende una petición lo primero que debe hacer es identificar su procedencia para poder analizar los datos recibidos de forma adecuada. Cada conexión recibida se atiende como una petición, que consta de una serie de datos en forma de ristra de bytes. Estos datos deben ser analizados sabiendo previamente cuál es su formato. Incluyen siempre el número de petición, el estado en el que se encuentra, que hace referencia al tipo de operación a realizar, y el resto de la información son datos que se usan para consultar o insertar información en la base de datos a través de su gestor, el *gestor de la base de datos*.

Su función principal es analizar el estado en que se encuentra la petición y, según cuál sea, realizar la operación oportuna. Esto implica descifrar la información que llega a partir del *socket* y ver el tipo de operación que debe realizar. Cuanto se trata de peticiones que implican la generación de plantillas las operaciones que puede realizar son las siguientes:

- Obtener la lista de dispositivos disponibles y algunas de sus características que soporta como los distintos tamaños y tipos de papel y los tipos de resolución. La lista de dispositivos y los ficheros XML correspondientes a sus capacidades los solicita al gestor de la base de datos. El *gestor XML* se encargará de obtener de los ficheros las listas de valores de las capacidades solicitadas.
- Obtener los datos generales que definen la plantilla y organizarlos en una estructura de datos para que el *gestor de base de datos* los almacene en la tabla *T_PLANTILLA*.
- Obtener todos los elementos que forman la plantilla junto con sus propiedades para organizarlos en estructuras correspondientes a cada tipo de dato. Toda esta información se almacena en la base de datos en las tablas correspondientes a los elementos que comienzan por *ELEM_*.
- Enviar a la *interfaz* toda la información referente a una plantilla seleccionada por el usuario para que se pueda visualizar. Se solicita la información al *gestor de la base de datos* para que consulte todas las tablas relacionadas con la plantilla.

Las peticiones de impresión implican operaciones similares pero relativas a plantillas, documentos y dispositivos. Estas operaciones son:

- Recoger los datos variables del documento o documentos que hay que generar junto con su nombre, número de copias y las opciones de impresión. Toda la información relativa a los documentos la recoge el *gestor de la base de datos* para almacenarla en la tabla *INFO_DOC*.
- Obtener a partir del nombre y código del dispositivo su estado y localización, así como todas sus capacidades para enviarlas a la interfaz.

Hay otras operaciones que realiza para todo tipo de peticiones:

- Comprobar la identificación de los usuarios con la información obtenida de la base de datos y si es correcta obtener su lista de plantillas generadas.
- Identificar el tipo de petición para ver si se trata de un *telegrama* o de una petición que llega desde la *interfaz* y asignarle un número que la identifique.
- Comprobar el estado de las peticiones y los datos obtenidos para ver si son correctos y en caso de no serlo comunicárselo a la *interfaz*.

En el caso de las peticiones que llegan a través de la base de datos, el *telegrama* contiene toda la información necesaria para la impresión del documento o los documentos. Por tanto, se realiza la obtención de todos los datos a partir de una cadena con formato predefinido para cada plantilla, según el tipo de telegrama que tenga asignado. El método para obtener los valores de todos estos elementos que componen la cadena es similar al de la función *scanf ()* del lenguaje C. La posición que ocupan estos datos en el documento se reconoce a partir del nombre asociado al elemento de texto en el momento de crearlo o bien por el orden de aparición en la plantilla creada, desplazándose de izquierda a derecha y de arriba abajo. A medida que se vayan obteniendo todos los datos se van organizando en forma de estructuras que comparte

con el *gestor de base de datos* para que pueda almacenarlas. Una vez almacenados estos datos, el *motor de impresión* se encarga de completar la petición.

Una vez explicado el último de los módulos de la parte principal de la aplicación, se va a describir el funcionamiento de la *interfaz*.

3.3.5 Diseño de la interfaz

En esta sección se describe el diseño de la *interfaz* que sirve para dar soporte a los usuarios. En un principio, se pensó en prescindir de esta parte y suplir su función mediante la creación manual de las plantillas. En vista de que el esfuerzo requerido para ello iba a ser grande, se decidió automatizar este proceso de creación junto con otra serie de opciones que resultaban interesantes. Se pretendía crear una interfaz manejable, sencilla e intuitiva para los usuarios. Esto no fue posible debido a cierta terminología de carácter técnico que se utiliza en algunas opciones que hacen referencia a los dispositivos y a la dificultad que presenta Java - lenguaje en que se ha desarrollado - para realizar ciertas funciones gráficas. Dado que su función es un mero soporte a la parte principal de la aplicación, no supone un problema y se define como una interfaz destinada a usuarios con ciertos conocimientos técnicos.

El diseño de esta parte de la aplicación consta de dos capas: una capa lógica y una capa de presentación. Para mostrar la información necesaria en cada momento la capa lógica se compone de los *gestores de ventana* que se representaban en la figura 3.2. Son una serie de clases que se utilizan para intercambiar datos con el *gestor de entradas* a través de *sockets*. Estas clases son creadas y llamadas desde las distintas *ventanas* que contiene la capa de presentación, que además invocan sus métodos a medida que necesitan obtener cierta información. En la figura 3.6 se muestra un diagrama de clases que contiene todas las que intervienen en esta parte de la aplicación.

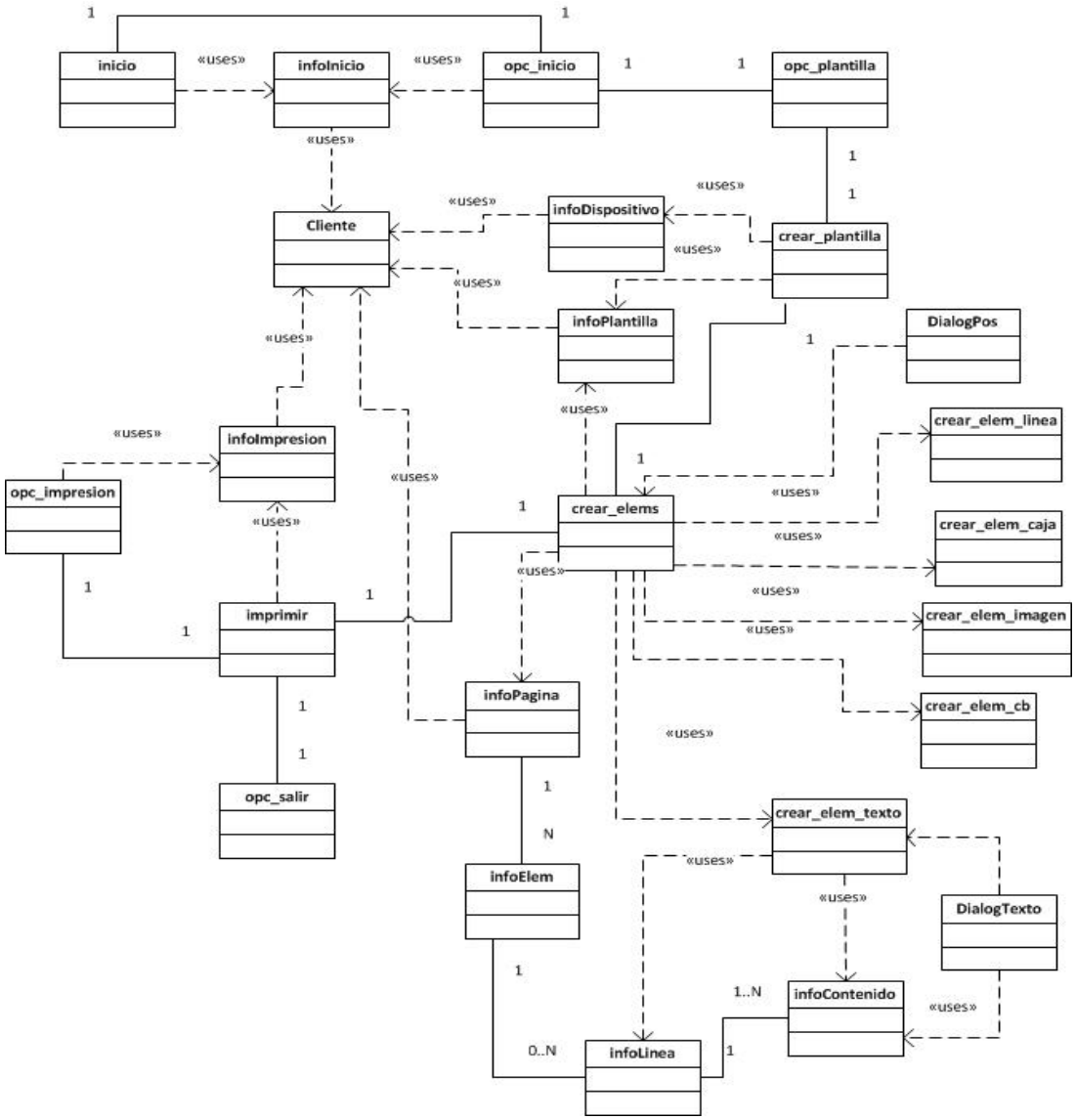

Figura 3.7 Diagrama de clases de la interfaz.

Todas las clases que se describen a continuación crean un objeto de clase *Cliente* para que establezca una conexión, basada en *sockets*, con el *gestor de entradas*. Su función es enviar y recibir cadenas de datos en forma de bytes. Los datos que envía son cadenas con un formato específico que obtiene del *gestor de ventana* que lo ha creado. Los datos que recibe del *gestor de entradas* se los envía al *gestor de ventana* y en el momento que se dé por finalizada la operación, se cerrará la conexión mantenida entre el *Cliente* y el *gestor de entradas* y el *Cliente* terminará su ejecución.

Los *gestores de ventana* que aparecen en la figura 3.7 son:

- *infoInicio*: es la clase encargada de gestionar toda la información relativa al usuario en el momento de su identificación y toda la que se recoge en la ventana *opc_inicio* de la aplicación *front_end.* Debe solicitar al *gestor de entradas* la lista de plantillas del usuario, la lista de dispositivos y sus opciones de impresión y configuración. Se

encarga de recoger los datos de entrada de los documentos a imprimir, así como el nombre de la plantilla y del dispositivo.

- *infoDispositivo*: es la clase que gestiona la información de cada dispositivo perteneciente a la lista de dispositivos obtenida por la clase *infoInicio*. Su información incluye el nombre del dispositivo, el estado, la localización, su lenguaje de impresión y un conjunto de listas que hacen referencia a opciones de impresión necesarias para la creación de las plantillas. Estas opciones son relativas al tamaño y tipo de papel, la resolución, el conjunto de caracteres y las fuentes de texto soportadas, ...etc.

- *infoElem*: es la clase que gestiona toda la información relativa a los elementos que contiene la plantilla. Permite la creación y edición de todos los tipos de elementos soportados por la aplicación. Todos los elementos tienen características comunes relativas a la posición que ocupan y sus dimensiones, pero también otras que son propias de cada tipo. Toda la información que guardan sobre los elementos la obtienen de las ventanas: *crear_elem_texto, crear_elem_imagen, crear_elem_linea, crear_elem_caja* y *crear_elem_cb*. Los elementos de texto utilizan las clases *infoContenido* e *infoLinea* para completar su gestión. Esto es debido a que este tipo de elementos puede estructurarse en una o varias líneas que pueden contener uno o más datos de texto. Esta estructura resulta útil cuando se quieren definir varias líneas de texto con datos separados por una gran cantidad de espacios. La clase *infoContenido* es la que gestiona la información de cada uno de los datos de texto que se crean en una línea perteneciente al elemento de tipo texto. Esta información la obtendrá de la ventana *DialogTexto* y hará referencia a su posición y dimensiones dentro de la página de la plantilla, al tipo de dato (fijo o variable) y a su contenido, en caso de ser fijo, o a su longitud, en caso de ser variable. La clase *infoLinea* se ocupa de gestionar la información de cada línea que forma el elemento de texto. Para ello generará una lista de elementos de tipo *infoContenido* ordenados por su posición dentro de la línea.

- *infoPagina*: es la clase encargada de gestionar la información de las páginas que forman la plantilla. Se ocupa de comprobar los límites de los elementos para que no se superpongan unos con otros y de guardar los elementos que se vayan creando, generando una lista de objetos *infoElem* por cada página.

- *infoPlantilla*: es la clase que gestiona toda la información relativa a las plantillas. Se encarga de guardar los datos de configuración de la plantilla y junto con las clases *infoPagina* e *infoElem* va organizando los elementos de la plantilla que se vayan creando. Al ser la encargada de guardar toda la configuración de la plantilla y sus elementos es la que se ocupa de enviar los datos al *gestor de entradas* cuando se termina la creación de la plantilla. En caso de que la operación a realizar sea la edición de plantillas existentes, será el *gestor de entradas* el que le envíe la

información de la plantilla para que pueda modificar su configuración. Las ventanas que utilizan este gestor de ventana son: crear_plantilla y crear_elems.

- *infoImpresion*: es la clase cuya única función es gestionar las opciones de impresión del dispositivo elegido. Es la encargada tanto de solicitar al *gestor de entradas* las opciones de impresión del dispositivo, como de recoger los valores seleccionados por el usuario para la impresión del documento y enviarlos de nuevo al *gestor de entradas*.

  Una vez definidos los elementos de esta capa lógica, se describe la capa de presentación. Primero se muestra su mapa de navegación y a continuación una serie de prototipos de las ventanas que componen la interfaz. El diagrama que representa el mapa de navegación se puede ver en la figura 3.8, formado por una serie de recuadros que corresponden a las ventanas y unidos por unas flechas que indican cómo se navega de una ventana a otra.

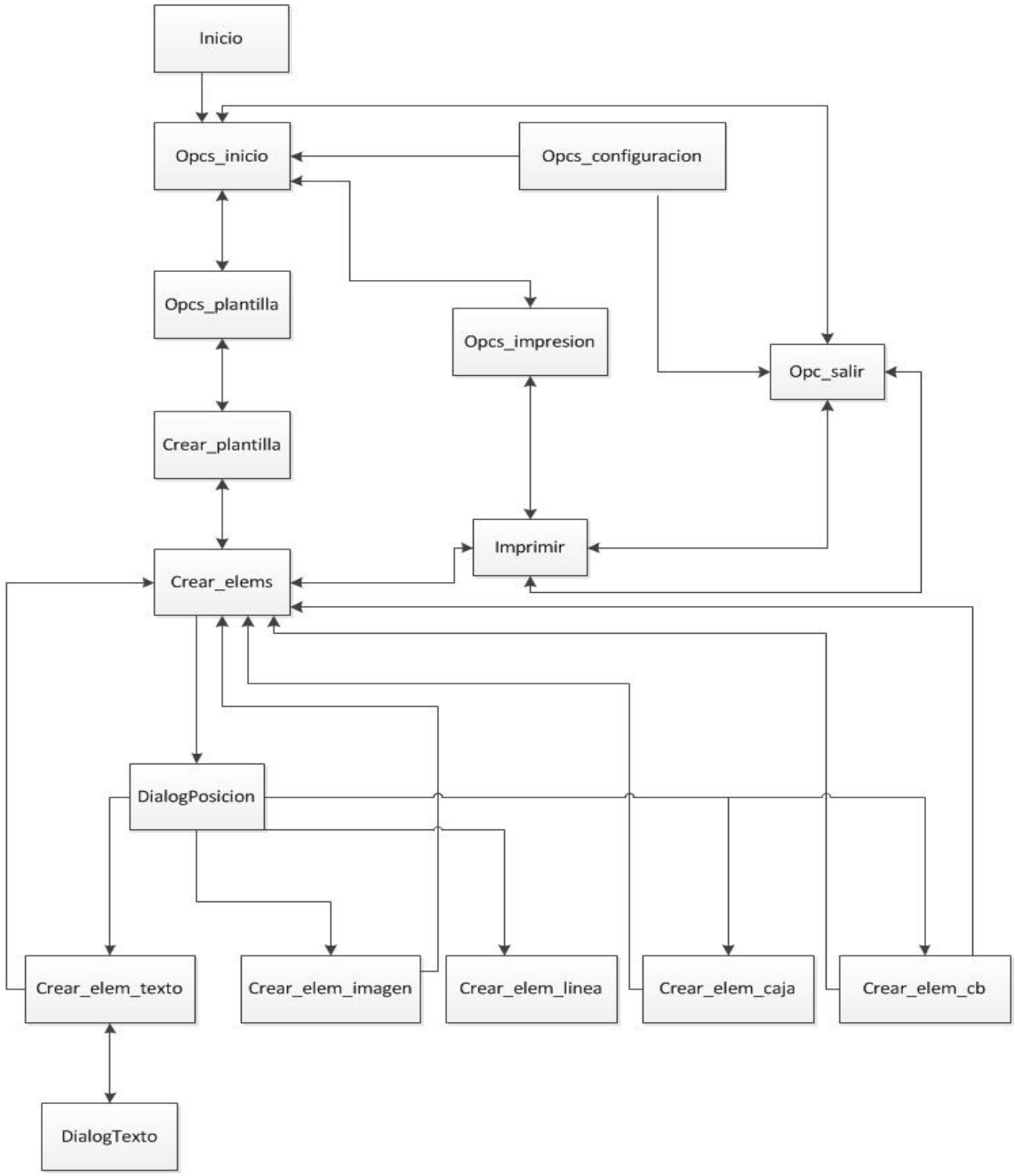

Figura 3.8 Diagrama del mapa de navegación de la interfaz diseñada

A continuación se presentan algunos de los prototipos a partir de los que se modelaron las ventanas principales que se incluyen en la interfaz final. El número de ventanas implementadas es numeroso y sólo se muestran los más destacados. Las ventanas se han desarrollado utilizando el IDE de la aplicación *Java Net Beans*. Los prototipos se pueden visualizar en la figuras 3.9, 3.10, 3.11, 3.12 y 3.13. Desde la figura 3.12 se ven algunas de las capturas realizadas a las ventanas en una de las pruebas de la interfaz.

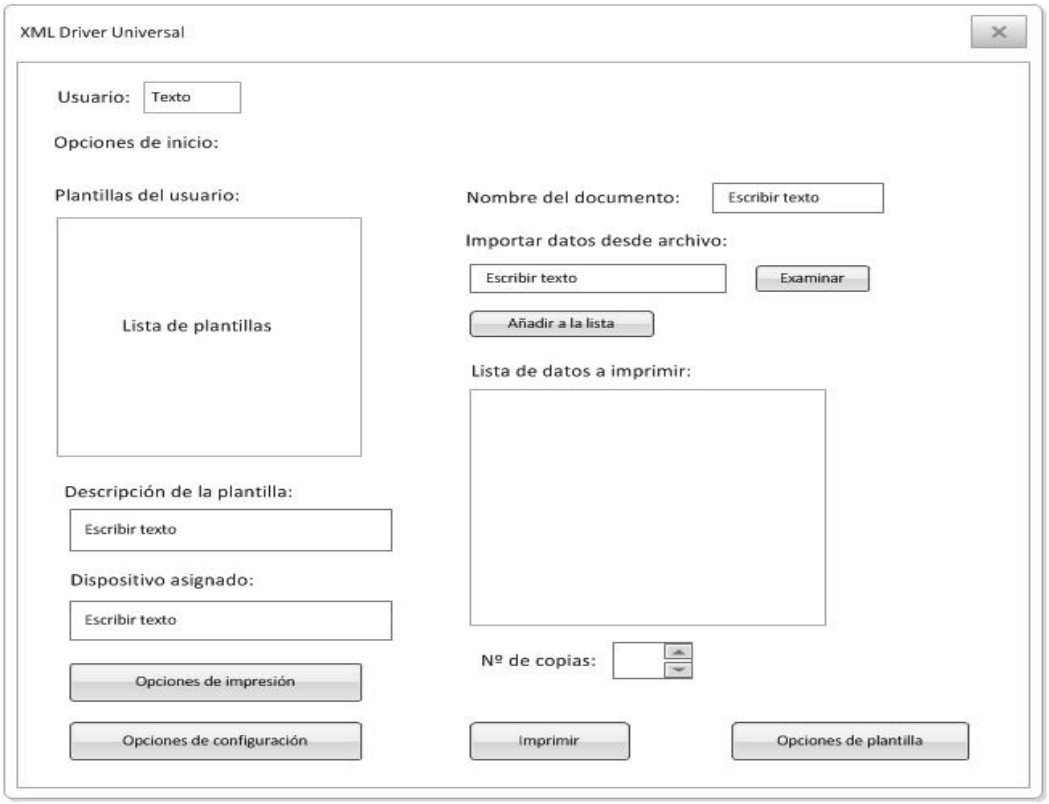

Figura 3.9 Prototipo de la pantalla de inicio de la aplicación.

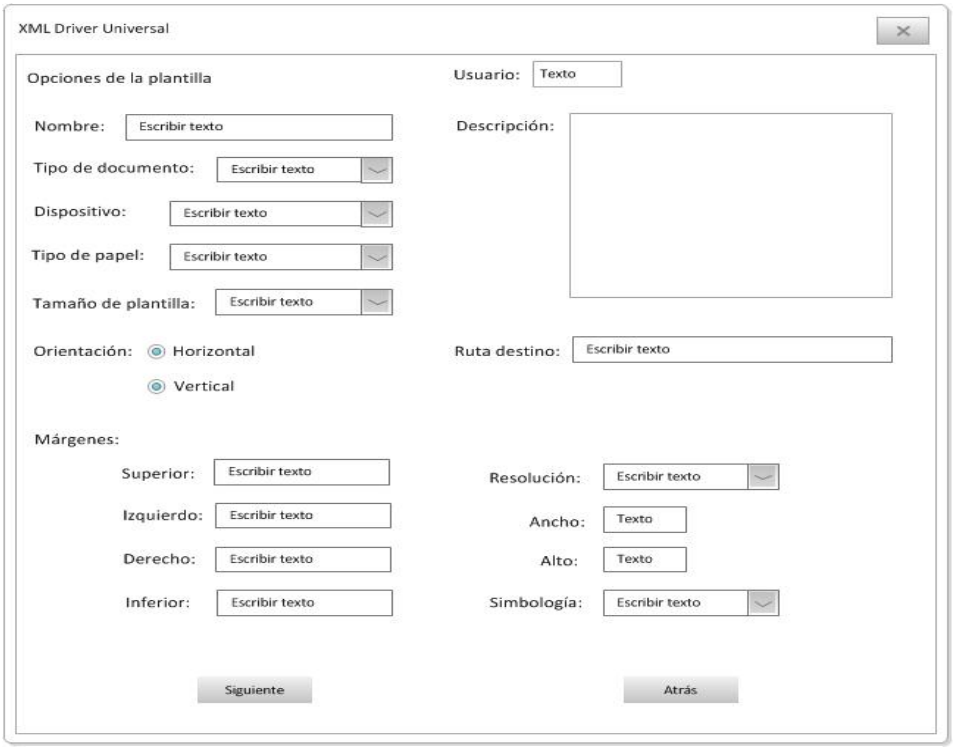

Figura 3.10 Prototipo de la pantalla de creación de plantillas.

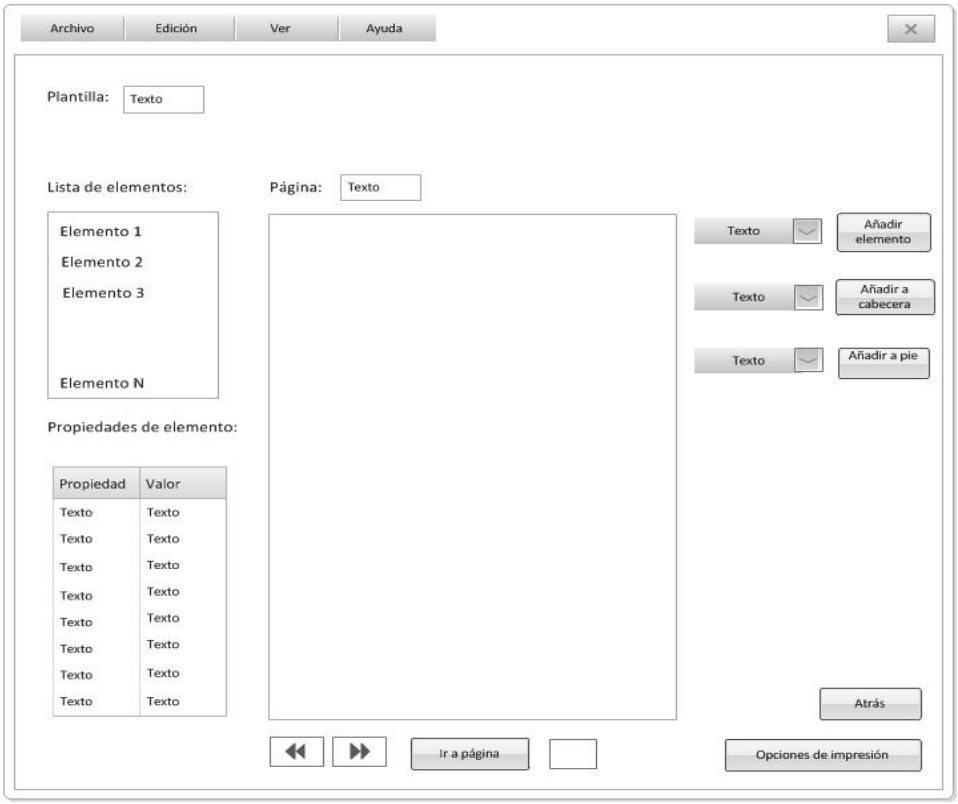

Figura 3.11 Prototipo de la pantalla de creación de elementos de la plantilla.

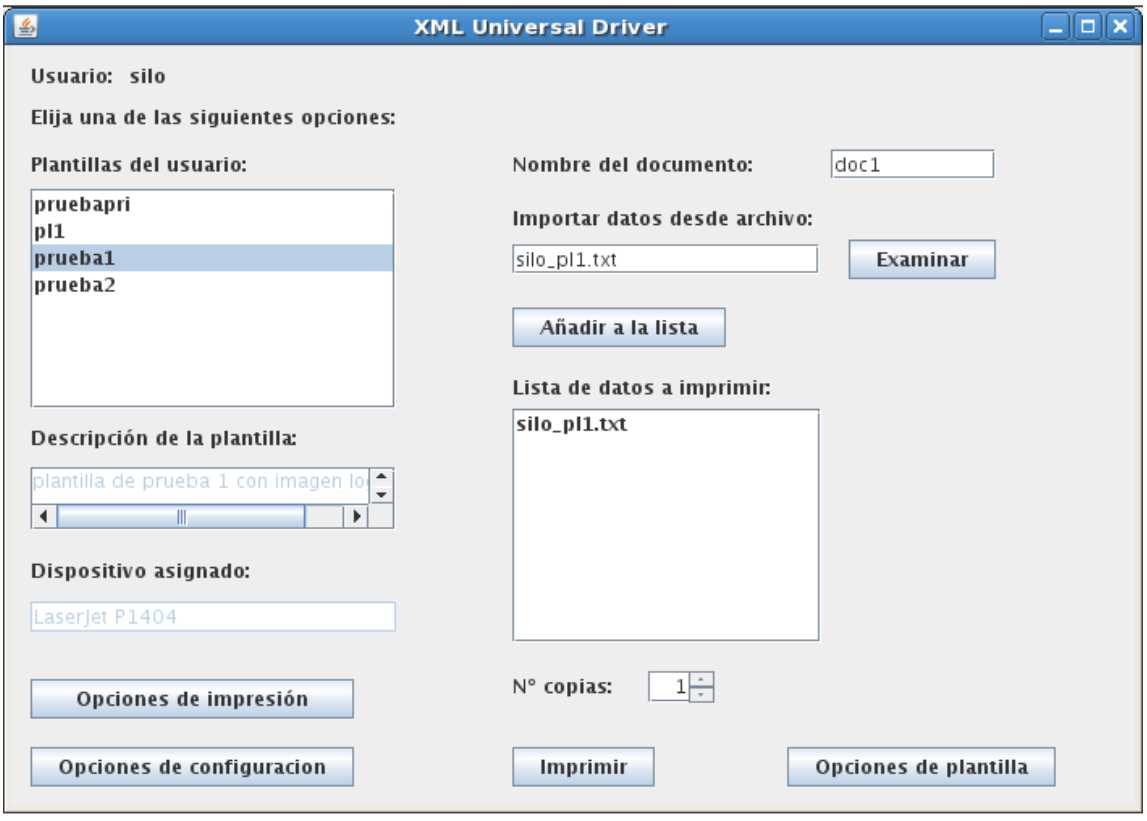

Figura 3.12 Pantalla con las opciones de inicio de la interfaz.

El prototipo de la figura 3.9 corresponde a la ventana de la figura 3.12. Como se puede ver apenas se aprecian diferencias. Esta ventana es *Opcs_inicio*, tal y como aparecía en el mapa de navegación de la figura 3.8, y es la primera que aparece una vez que el usuario se identifica al inicio de la aplicación. La lista que aparece a la izquierda muestra las plantillas que el usuario ha creado previamente y al seleccionarlas se da una descripción de la plantilla y del dispositivo que tiene asignado. Se ofrecen una serie de opciones por medio de botones para gestionar las plantillas, imprimir documentos y ver las opciones de impresión y configuración.

La figura 3.13 muestra la ventana *Crear_plantilla*, cuyo prototipo se podía ver en la figura 3.10. Esta ventana presenta la información general de la plantilla que el usuario deberá rellenar. Se muestra la lista de dispositivos disponibles para asignarle uno de ellos a la plantilla. Al seleccionar uno de los dispositivos aparecerán las listas de algunas de las opciones que soporta: tamaños de página, tipos de papel, resolución y simbología. Cuando el usuario pulsa el botón "Siguiente" se guarda la configuración inicial de la plantilla y se carga la ventana *Crear_elems* para poder añadir nuevas páginas y elementos a la plantilla.

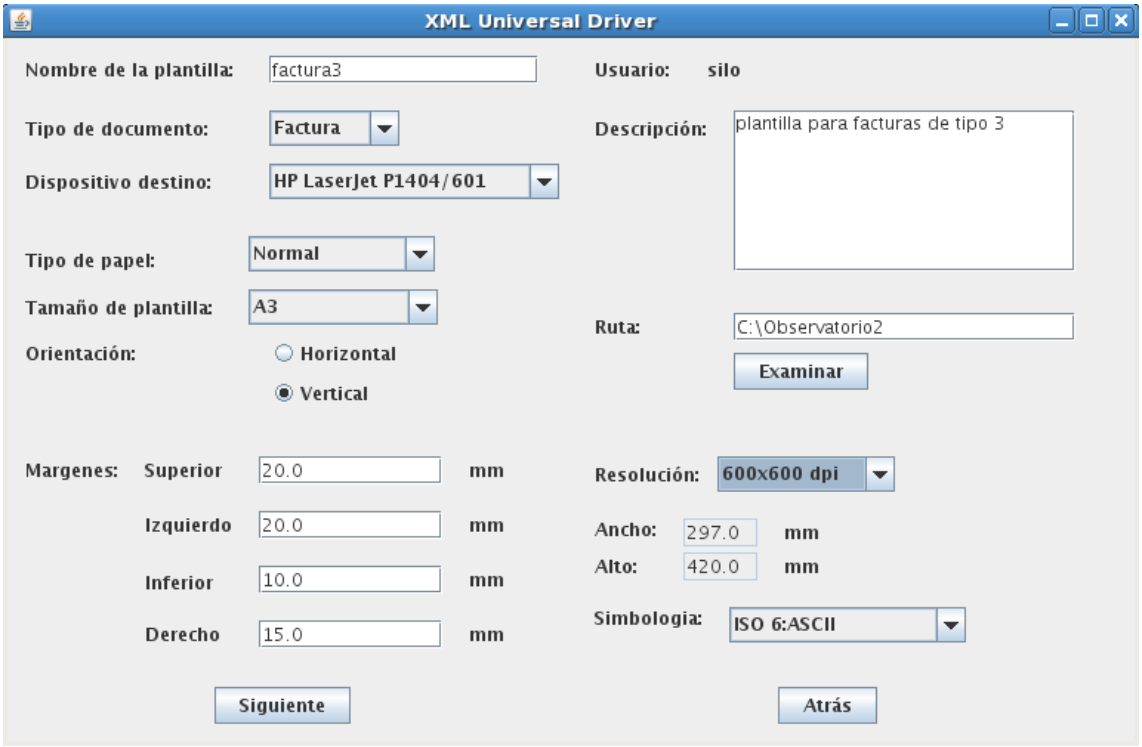

Figura 3.13 Pantalla que inicia la creación de plantillas

Capítulo 4

Gestión de proyecto

En este capítulo se describe cómo ha sido la gestión del proyecto. La sección 4.1 analiza la metodología de desarrollo seguida, cuyo objetivo es presentar las técnicas que han permitido modelar el sistema. En la sección 4.2 se muestra la planificación del proyecto fase por fase y las tareas realizadas en cada una de ellas. Se proporcionan diagramas en los que se puede visualizar la duración estimada para las distintas tareas. La última sección de este capítulo resume el esfuerzo dedicado al proyecto en cada una de las fases que lo componen.

4.1 Metodología

La Metodología de desarrollo de software utilizada toma como referencia el Proceso Unificado [19] en el que se basa UML (*Unified Modeling Language*, por sus siglas en inglés). Es el lenguaje de modelado de sistemas de software más conocido y utilizado en la actualidad. Es un lenguaje gráfico para visualizar, especificar, construir y documentar un sistema. Ofrece un estándar para describir el modelado de un sistema, incluyendo aspectos conceptuales tales como funciones del sistema y aspectos concretos como expresiones de lenguajes de programación, esquemas de bases de datos y componentes reutilizables. Para tal fin se describen herramientas de Análisis y Diseño Orientado a Objetos, diagramas, especificación, y criterios de aplicación de las mismas. Su objetivo es presentar un conjunto de técnicas tradicionales y modernas de modelado de sistemas que permitan desarrollar software de calidad. Es importante resaltar que UML es un "lenguaje de modelado" para especificar o para describir métodos o procesos. La decisión de basarnos en esta metodología es porque es aplicable a muchos proyectos de ingeniería del software y ofrece una visión del sistema basada en módulos que resulta más sencilla en comparación con otras metodologías basadas en procesos. La metodología seguida no se puede considerar UML al 100% ya que la terminología utilizada en los gráficos y diseños no cumple con el estándar, pero éste ha sido tomado como referencia. No se proporciona una notación tan clara y sencilla como la que ofrece UML, pero se ha intentado simplificar todo lo posible para que los diseños fueran claros a pesar de la dificultad y extensibilidad que entraña el proyecto.

El ciclo de vida seguido durante este proyecto ha sido secuencial entre las distintas fases e iterativo dentro de alguna de ellas. Aunque en la primera etapa de desarrollo no se obtiene una reducción en el tiempo, las iteraciones subsiguientes son más rápidas y más fáciles que si utilizamos un enfoque convencional, puesto que las revisiones a realizar

están mucho más localizadas. Además de todas las actividades correspondientes a cada fase, que se van a describir en la siguiente sección, hay una serie de funciones de soporte del proyecto, que se prolongan a lo largo de todo su ciclo de vida. Estas funciones son:

- Gestión del proyecto: engloba todas las actividades de gestión del proyecto y sirve como guía para recopilar todo lo que hay que tener en cuenta. La gestión de la planificación realizada se incluye dentro de estas funciones.

- Gestión de configuraciones: define los procedimientos y políticas a seguir en ciertas actividades (nombrado de documentos, control de cambios, gestión de versiones, etc.), así como las herramientas a ser utilizadas. Parte de esta gestión viene determinada por la que se realiza en la empresa HP.

- Gestión de la calidad: establece procedimientos y estándares de calidad aplicables al proyecto y verifica su aplicación. Se define un plan de aseguramiento de la calidad con actividades aplicables a cada fase del desarrollo del proyecto. En este proyecto no se ha definido un plan específico para asegurar la calidad.

- Verificación y validación: verifica que el producto final hace lo que debe y lo hace bien. Para ello se realiza un seguimiento de las pruebas y se muestra constancia de su realización. Se trata de definir una serie de casos de prueba y realizar un informe con los resultados obtenidos en cada una de las pruebas.

4.2 Planificación

La planificación consta de una serie de fases que se llevan a cabo de forma secuencial a lo largo del tiempo, realizando algunas de sus tareas de forma solapada. Al principio del proyecto, y con la ayuda del director, se programó una planificación inicial basada en su experiencia, que, por causas que se detallan a continuación, no fue posible cumplir. Se van a detallar punto por punto cada una de estas fases y en la figura 4.1 se puede ver una comparativa entre la planificación inicial y final. La planificación inicial se representa con las barras que tienen color naranja y la planificación final con color azul. Cada barra se corresponde con una de las cinco fases en las que se ha divido el proyecto, que aparecen a la izquierda en la figura 4.1. Estas fases son: fase de investigación y análisis, fase de diseño, fase de construcción y elaboración de prototipo, fase de evaluación y fase de documentación. Cada una de estas fases incluye una serie de tareas más concretas, pero por falta de visibilidad no se han incluido en la figura 4.1. En este diagrama no se observa, pero la dedicación al proyecto era de lunes a viernes con una media de 6 horas al día. Se planificó el tiempo que debía ocupar cada fase del proyecto y se fue recopilando información de lo que se había hecho y lo que estaba en progreso en ese momento. De esta forma se podía ver el estado del proyecto cuando fuera necesario y planificar el uso de los recursos. Se han utilizado gráficos Gantt como

herramientas de planificación para llevarla a cabo. No se muestran estos gráficos porque debido a la larga duración no se podían distinguir bien las fases y tareas.

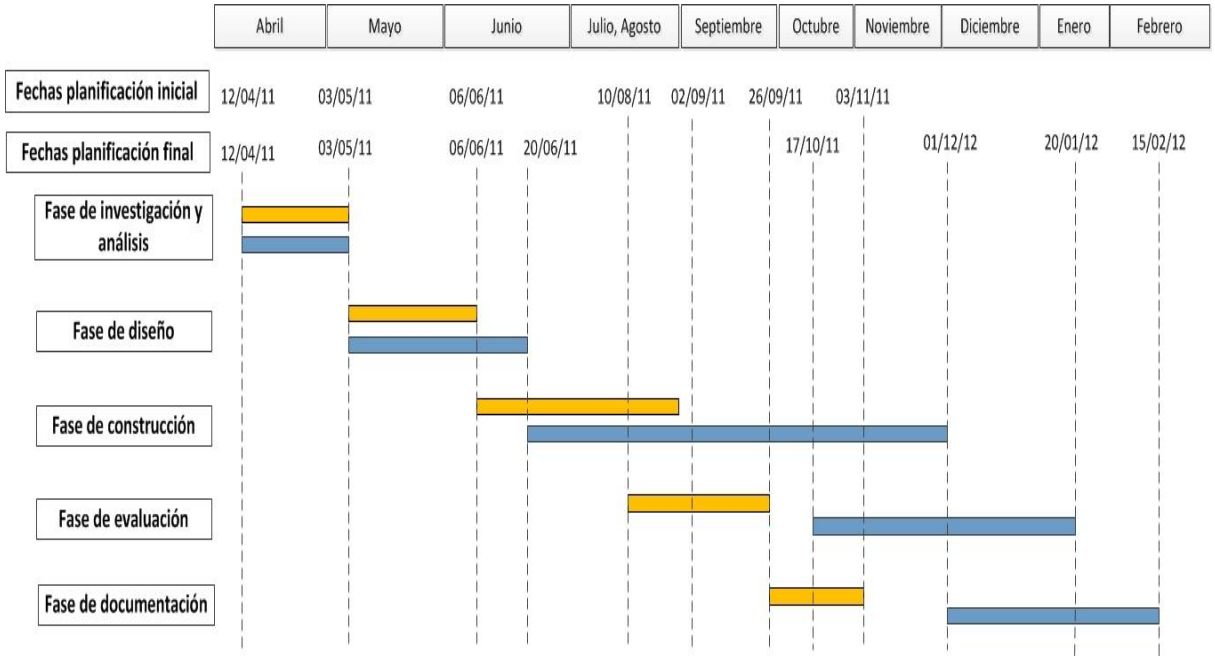

Figura 4.1 Planificación inicial y final del proyecto

A medida que el proyecto fue adquiriendo cierta envergadura, era necesario realizar fases de iteración sobre los elementos que lo componían, ya fuera por la ocurrencia de cambios en los requerimientos del proyecto o por la obtención de mejoras. Las fases donde se producen estas iteraciones principalmente son: la fase de análisis por la adición de nuevos requisitos y la fase de construcción por la implementación de cambios y mejoras en la generación del código. A lo largo de todas las fases se mantuvieron reuniones tanto con el ponente como con el director del proyecto, ya fuera en EINA o en HP.

Se puede observar en la figura 4.1 el gran desfase de tiempo que existe entre las planificaciones inicial y final, que comienza levemente en la fase de diseño y se dispara en la fase de construcción y elaboración de prototipo. En la gestión del proyecto se deben identificar los posibles riesgos, generando un estudio de los mismos, así como su impacto estimado y real. Debido a la falta de experiencia en la planificación no fue así y se cometieron errores. Aún con la ayuda del director, no se pudo prever la dificultad que supondría aprender las nuevas tecnologías utilizadas, cuyo comienzo fué al final de la fase de diseño. Otros factores que influyeron fueron las vacaciones de verano y las de Navidad, en las que no hubo apenas reuniones y se disminuyeron los esfuerzos por la falta de continuidad en la supervisión de tareas. Por último, el acceso restringido a las instalaciones de HP no ayudó a la finalización del proyecto a tiempo.

4.2.1 Fase de investigación y análisis

En esta primera fase se trató de recopilar toda la información posible del proyecto de cara a obtener un Plan detallado de proyecto. A partir de este punto comenzó una etapa de investigación sobre una parte representativa de los productos de impresión / etiquetación / rotulación existentes en el mercado y se llevó a cabo un estudio de sus características y funcionalidades. Una vez finalizado dicho estudio se procedió a realizar el análisis (análisis de requisitos y plan de pruebas) y diseño (diagramas de secuencia, de actividades,…) -mediante metodologías de desarrollo como UML. De esta forma los elementos fueron integrados progresivamente y resultó más fácil ir detectando posibles errores. Esta fase podía realizarse en cualquier ámbito donde se dispusiera de un ordenador y acceso a Internet para buscar la información necesaria. Se consultó con el director cualquier duda que pudiera surgir, teniendo reuniones semanales o enviando correos.

Las tareas fundamentales en esta fase fueron:

- Análisis de requisitos y modelado: obtención de requisitos funcionales, no funcionales y de facilidad de uso, así como el modelado de los casos de uso y la arquitectura. Los entregables producidos por esta tarea son:
    - Objetivos generales del proyecto y especificación de requisitos (funcionales y no funcionales): además de los objetivos y el alcance del proyecto, se enumeran una serie de requisitos que debe cumplir el sistema, dando una visión de alto nivel del proyecto. La mayoría serán obtenidos en la etapa de investigación mediante la recopilación y organización de la información.
    - Análisis de alternativas de solución: describe distintas soluciones que pueden haber surgido en esta primera etapa y se justifica la elección de la solución adoptada.
    - Análisis de tecnologías a utilizar: proporciona una descripción de las distintas tecnologías en las que se apoya el producto software para su funcionamiento.
- Especificación de las pruebas de aceptación: establece una serie de pruebas a ejecutar por el usuario, que servirán para comprobar el correcto funcionamiento del sistema respecto a los requisitos especificados por el desarrollador del mismo.

4.2.2 Fase de diseño

En esta fase se realiza el modelado del sistema mediante distintos diagramas (de secuencia, de componentes, de despliegue,…), así como posibles patrones de diseño y el diseño de interfaces. Los entregables producidos por esta tarea son:

- Especificación de la arquitectura del software: define la arquitectura modular del sistema describiendo los módulos principales que la componen y las relaciones que existen entre ellos. Así mismo, se pueden especificar las funcionalidades principales de cada módulo de forma independiente. Se da una visión a nivel arquitectural y también a nivel de componentes, tanto estática como dinámica.

- Especificación del diseño: comprende el modelo de objetos (clases, BBDD,…) y las estructuras de datos que usa la aplicación, dando una visión del sistema más detallada y de más bajo nivel. El diseño de interfaces debe incluir un mapa de navegación, que muestre las comunicaciones existentes entre las distintas ventanas de la aplicación, y un prototipo de ventanas, que muestre la forma en la que se van a presentar las ventanas de la aplicación.
- Especificación de las pruebas unitarias: estas pruebas se realizan sobre cada módulo de manera individual y deben cubrir todos los puntos de test necesarios para la detección de posibles errores. Para ello se pueden usar métodos de caja blanca y de caja negra o incluso combinarlos.

- Especificación de las pruebas de subsistemas y del sistema: se definen pruebas de integración entre los distintos módulos, así como pruebas de validación y pruebas del sistema. Las pruebas de validación revisan los requisitos establecidos y comprueban que todas las funcionalidades especificadas aparezcan en la aplicación y se comporten de la forma esperada.

4.2.3 Fase de construcción y elaboración de prueba de concepto-prototipo

Esta fase comenzó con la elaboración de un prototipo, cuyo objetivo fue probar que el concepto elaborado en la fase anterior podía ser llevado a cabo funcionalmente. En un principio, se estimó que su duración iba a ser de un mes y era suficiente con que tuviera una funcionalidad mínima para ser probado. No fue posible terminarlo en el plazo estimado debido al desconocimiento de las tecnologías que se iban a utilizar y al lenguaje de programación Java, del que apenas se tenía experiencia al programar. Se tomó la decisión de elaborar el prototipo empleando los mismos lenguajes de programación que iban a ser usados en la fase de implementación. De esta forma se podría aprovechar todo el código que iba a ser generado y no supondría mayores retrasos. Hasta ahora se habían cumplido las estimaciones de tiempo planificadas, pero a partir de este punto no se pudo seguir la planificación diseñada inicialmente. A continuación se inició la implementación del sistema. Al mismo tiempo que se implementaban los diferentes módulos que iban a componer el sistema, se iban realizando test para cada uno de ellos. Posteriormente se fueron integrando todos los

módulos haciendo los test definidos en la fase anterior. Se dispusieron las herramientas necesarias en cuanto a software, para realizar la codificación, así como los equipos necesarios para ir realizando las primeras pruebas. No fue posible realizarlo en las instalaciones de HP, lo cual ha influido negativamente en el desarrollo de esta fase debido a la imposibilidad de acceder a la aplicación SILO como un usuario autorizado. Este hecho supuso la instalación de una máquina virtual en un ordenador portátil para uso personal, proporcionado por el BIFI, cuyo objetivo era recrear el entorno de la aplicación SILO. Se instaló el sistema operativo *Red Hat 6.0* de *Linux* y el gestor de base de datos *Oracle*, en su versión 10.0. Una vez creado el soporte para la base de datos, se realizó una copia de las tablas principales de SILO, junto con los datos que contenían.

En esta fase hay dos entregables:

- Prueba de concepto/prototipo; a partir de este punto surgen ciertas correcciones a los parámetros iniciales de diseño para pasar al siguiente entregable.
- Construcción de entregables software finales y documentación (de los entregables), pruebas, especificaciones, parámetros, etc.

4.2.4 Fase de evaluación

La fase final del proyecto debe ser de verificación y validación del mismo. Incluye la documentación de las pruebas elegidas con el fin de valorar la aceptación del sistema y su idoneidad y capacidad de integración. En la etapa de análisis o en su defecto en la de elaboración de la prueba de concepto-prototipo se ha definido la metodología a seguir en las pruebas y en esta etapa se realizan las mismas y documentan sus resultados. Todos los errores encontrados deben ser corregidos. En esta etapa del proyecto se dan por finalizadas todas las fases. Las pruebas se realizaron en las instalaciones de HP, que es donde se encontraban los dispositivos adecuados para ello.

Los entregables producidos en esta etapa de pruebas son los resultados de las mismas y se detallan en: el informe de las pruebas unitarias, de las pruebas de subsistemas, de las pruebas del sistema y de las pruebas de aceptación.

4.2.5 Fase de documentación

En esta fase se genera la memoria final del proyecto, que engloba formalmente todo lo que se ha ido haciendo a lo largo del proyecto, incluyendo todos los entregables citados en las distintas fases del proyecto. La parte principal contiene un resumen de unas 20 hojas donde se detallan los aspectos fundamentales del proyecto y se le añaden una serie de anexos que contengan:

- Documentación técnica: esquemas, diagramas, modelos, prototipos, etc.

- Manual de usuario: proporciona las instrucciones a seguir para la utilización del producto.
- Manual de instalación: proporciona las instrucciones a seguir para la instalación del producto.
- El código fuente de los objetos de intercambio entre la aplicación y los elementos de impresión, que se desarrollará preferiblemente en Java. Por otro lado también se proporcionan los objetos necesarios para obtener los esquemas XML/XSL.
- Los scripts necesarios para la compilación y ejecución del sistema.

4.3 Esfuerzo real dedicado

El proyecto comenzó realmente con la fase de investigación el 12 de Abril de 2010. Previamente durante el mes de Marzo hubo un par de reuniones con el director y algunos contactos del BIFI para hablar sobre la finalidad del proyecto y posteriormente con el ponente también. La duración del proyecto se estimó en unos seis meses, que correspondía a unas 500 horas. Se estableció como fecha límite de entrega el mes de Noviembre de 2011 para su presentación en el mes de Diciembre del mismo año. No fue posible realizar su entrega a tiempo y se pospuso al mes de Febrero de 2012, presentándolo en Marzo de 2012.

Se puede ver el cambio de planificación realizado en la figura 4.1, lo que influye directamente en el aumento del esfuerzo dedicado al proyecto. En total, el número de horas dedicadas al proyecto ha sido de 849. Esto supone que esfuerzo real dedicado no se corresponde con el de la planificación inicial y excede en 349 horas el número total previsto.

En esta sección se muestra en la figura 4.2 un diagrama con el tiempo invertido, en horas, en cada fase del proyecto. Además se acompaña de un porcentaje que denota mejor cuáles son las fases que mayor esfuerzo han requerido. Las fases de *construcción y elaboración de prototipo* y de *evaluación* se han unido para obtener sus horas totales, ya que están muy relacionadas y se produjeron iteraciones de reconstrucción de código según los resultados obtenidos en las pruebas. Como se puede observar en la figura 4.2 son las fases que mayor esfuerzo han requerido con una gran diferencia respecto al resto de fases. Como se comentaba en la sección anterior sobre los retrasos ocurridos durante el proyecto, esto se debe a que durante el comienzo de esta fase hubo que aprender nuevas tecnologías. La mayoría de las tecnologías usadas eran desconocidas y esto supuso una gran dedicación de tiempo durante la elaboración del prototipo o prueba de concepto. El resto de porcentajes de las otras fases se corresponden de una manera más adecuada con los que se suelen estimar en proyectos de ingeniería del software similares.

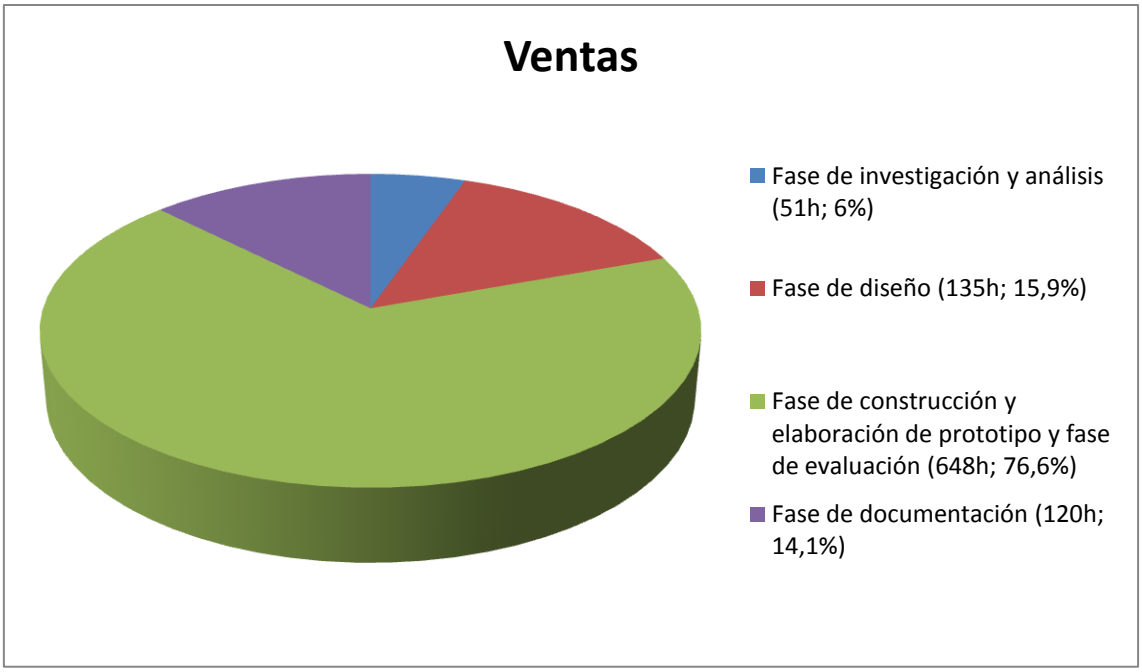

Figura 4.2 Diagrama con la relación de horas y porcentaje de esfuerzo dedicados al proyecto

Capítulo 5

Conclusiones

En este capítulo se exponen una serie de conclusiones finales a modo de culminación y cierre del proyecto.

5.1 Técnicas

Al finalizar este proyecto se han cumplido todos los objetivos principales que habían sido marcados en un principio. Se ha creado un driver universal basado en XML que es capaz de gestionar correctamente las peticiones de los usuarios a través de la interfaz y, a su vez, sincronizarse con otros procesos que requieran sus servicios. Se han puesto en práctica diferentes técnicas de programación en las que apenas se había profundizado en la carrera, ya que no se han construido aplicaciones de tal magnitud. Un ejemplo de esto fue la necesidad de comunicar la interfaz, que iba a ser implementada con el lenguaje de programación Java, y un servidor, cuya implementación estaba siendo realizada con el lenguaje C. En un principio se intentó utilizar la librería JNI (*Java Native Interface*) que ofrece Java, pero no dio resultado a pesar de consultar manuales, encontrados a través de Google, que indicaban su uso. Se optó por utilizar *sockets*, ya que proporcionaban una solución más independiente de los tipos de datos que manejan los distintos lenguajes.

Para poder desarrollar una solución al problema presentado fue necesario comprender el funcionamiento interno de los distintos dispositivos. Se visitaron las páginas Web de los fabricantes de los dispositivos y se obtuvieron los manuales de los lenguajes de impresión que se iban a utilizar para crear los *parsers* que generarían el código apropiado para una correcta impresión. Estos manuales contenían una gran cantidad de información referente al sistema de coordenadas usado para el posicionamiento de los datos a imprimir, las capacidades disponibles para los dispositivos que utilizaban dicho lenguaje y los comandos para seleccionar sus valores adecuados. Debido a la extensibilidad de los mismos sólo se tuvieron en cuenta sus características más importantes. La gestión de las imágenes y los códigos de barras ha sido la más compleja, ya que dependía del lenguaje de impresión empleado. Hubo que manejar distintos formatos para las imágenes y resultó ser un trabajo bastante costoso debido al desconocimiento de los mismos. Más complejo aún resultó el manejo de los códigos de barras. Esto fue debido a la cantidad de códigos de barras existentes en el mercado y las distintas características que presentan según su dimensión y tipo.

Respecto a la parte implementada en C, no se necesitó formación adicional para llevarla a cabo puesto que durante la carrera es uno de los lenguajes más utilizados. La implementación de funciones en SQL embebido en C tampoco fue un problema por el hecho de tener cierta experiencia previa en la realización de un trabajo universitario en el que era necesaria su utilización. Los principales problemas técnicos en esta parte de la aplicación surgieron al realizar las pruebas y fueron provocados por los datos de tipo cadena que son manejados, ya que en ocasiones daban valores nulos y provocaban fallos no controlados por el sistema.

5.2 Personales

Este proyecto ha aportado grandes experiencias personales. Se ha basado en la tecnología XML, que ha sido una de las más utilizadas en los últimos años y lo sigue siendo hoy en día. Se considera un estándar para el envío de documentos entre distintos dispositivos. Ha resultado interesante aprender esta tecnología y ver la cantidad de posibilidades que ofrece su uso. También se ha podido ver el potencial que ofrece XSLT, ya que de una manera sencilla permite la transformación de los documentos XML al formato de salida deseado. Aunque XML fuera la tecnología que diera nombre al proyecto debido a su importancia, se han aprendido muchas otras tecnologías durante el desarrollo del proyecto. Ha sido gratificante aprender nuevos conceptos referentes a los dispositivos de impresión y en general todos aquellos que incluye la creación de un driver. Los conocimientos adquiridos son raramente utilizados hoy en día, pero precisamente esa peculiaridad es la que propició la elección del proyecto. Por otro lado, la elaboración de la interfaz, aunque fue costosa, mereció la pena por el hecho de poder adquirir conocimientos acerca de uno de los lenguajes de programación que más se utilizan hoy en día debido a su portabilidad.

El hecho de que sea un proyecto bastante técnico, aunque no sea de una dificultad excesiva, proporciona una experiencia personal bastante completa al haber encontrado una solución al mismo y haber alcanzado los objetivos propuestos en un principio.

El motivo principal que impulsó la elección del proyecto fue el deseo de realizarlo en una empresa para tener un primer contacto con el mundo laboral. Este objetivo sólo se ha cumplido en parte debido a la imposición de un acceso restringido a las instalaciones de HP. En ese aspecto la experiencia no ha sido la esperada.

5.3 Trabajo futuro

Lo que aquí se ha creado es una amplia base que posteriormente permitirá ir añadiendo nuevas funcionalidades para incluir todo tipo de dispositivos, no sólo los de impresión. Se han abarcado las principales marcas de impresoras y etiquetadoras del mercado desarrollando los *parsers* de los lenguajes de descripción nativos más utilizados. Existen otros lenguajes, encontrados durante la fase de investigación, que podrían

haberse tenido en cuenta pero la comercialización de los dispositivos a los que corresponden es más reducida. Algunos de estos lenguajes podían ser añadidos gradualmente si fuera necesario.

En un principio se pensó que tendría interés investigar sobre los dispositivos móviles debido a la gran cantidad de mercado que mueven. Las funcionalidades necesarias para incluir este tipo de dispositivos no han sido desarrolladas y serían de gran utilidad si se añadieran en un futuro. En este caso su fin no sería ser impreso, sino ser visualizado en el dispositivo elegido como destino. Tratándose de un proyecto perteneciente al Departamento de Logística de HP, con esta nueva funcionalidad sería posible enviar cualquier documento a la pantalla de a bordo de los camioneros o a las pistolas de almacén de los empleados de una empresa de Logística.

Se podrían añadir mejoras a la interfaz diseñada, facilitando la accesibilidad a otro tipo de usuarios que no tengan necesariamente cierta formación técnica. Esto sería posible haciéndola más intuitiva a la hora de crear los elementos en la plantilla. Para ello habría que añadir nuevas funcionalidades gráficas que permitieran una selección directa de las posiciones mediante el uso del ratón y además añadir un zoom para ver una zona seleccionada con más detalle.

Al tratarse de un proyecto destinado a formar parte de la aplicación SILO de la empresa HP, seguro que se añaden nuevas funcionalidades en un futuro.

Anexo A

Análisis de tecnologías

En este anexo se explican las tecnologías utilizadas para desarrollar este proyecto. El objetivo es familiarizarse con la tecnología principal, que es XML y es la que permite la creación de archivos que contengan la información de impresión de documentos. Este apartado es importante para saber de qué se habla en cada proceso del desarrollo de la aplicación.

Únicamente se va a explicar la tecnología que se ha considerado necesaria para la realización del proyecto, sin extenderse con otras definiciones que no aporten nada.

Se estudian varias tecnologías:

- El lenguaje de datos XML y el de transformaciones XSLT, todo siguiendo las definiciones de la *W3C*.
- La tecnología Java.
- El servidor de bases de datos Oracle utilizando el lenguaje PL/SQL embebido en C. Analizaremos también el uso del lenguaje C.
- Los lenguajes de las impresoras: ZPL, PCL y PostScript.
- La sincronización entre procesos.

A.1 XML

XML, sigla en inglés de *Extensible Markup Language* («lenguaje de marcas extensible»), es un metalenguaje de etiquetas desarrollado por el denominado "*World Wide Web* Consortium" (W3C).

La base del XML es la introducción de datos estructurados en un fichero de texto mediante una serie de reglas, creando archivos fácilmente generados y leídos por un ordenador, cuya principal utilización consiste en permitir compartir estos datos a todos los niveles, por todas las aplicaciones y soportes. Es una tecnología cada vez más usada para el intercambio de archivos entre distintas plataformas debido a que es un estándar.

De la misma forma que el lenguaje HTML, el XML utiliza "tags" -etiquetas- y atributos para su definición, pero HTML combina la interpretación de los datos con la

presentación de estos, mientras que el lenguaje XML usa las etiquetas sólo para delimitar datos, y deja su interpretación, a la aplicación que los lee.

XML se escribe en un documento de texto ASCII, igual que el HTML la cabecera del texto se inicializa con el siguiente comando:

<?xml version="1.0"?>

En el resto del documento se deben escribir etiquetas como las de HTML: <ETIQ1>...<ETIQ2>...</ETIQ2>...</ETIQ1> , donde cualquiera de ellas puede tener atributos.

<ETIQ atributo1="valor1" atributo2="valor2"...>

Los comentarios de XML se escriben igual que los de HTML.

<!-- Comentario -->

Para definir qué etiquetas y atributos debemos utilizar al escribir en XML tenemos que fijarnos en la manera de guardar la información de una forma estructurada y ordenada.

Las reglas básicas del XML son:

- Un elemento root: Todo el documento XML debe estar contenido en un único elemento, al que llamaremos 'root'.
- Elementos anidados: Todos los elementos deben contener elementos completos, es decir, deben contener tanto la etiqueta de apertura como la de cierre. Por tanto se debe respetar el orden inverso al de apertura a la hora de cerrarlos.

  { <elemento1>…<elemento2>...</elemento2>…</elemento1> }
- Atributos entre comillas: Todos los atributos de los elementos deben tener su valor entre comillas. Pudiéndose elegir entre comillas simples o dobles.

  { <elemento atributo="valor"> = <elemento atributo='valor'> }
- Mayúsculas y minúsculas**:** Las etiquetas en XML, a diferencia con el HTML, diferencian entre mayúsculas y minúsculas.
- Cerrar elementos**:** Todos los elementos deben cerrarse con la etiqueta correspondiente.

  { <elemento>...</elemento> }
- Elementos vacíos**:** Los elementos que no contengan nada pueden abrir y cerrarse con una única etiqueta

  { <elemento/> }

A continuación se muestra como ejemplo un fichero XML:

```
<?xml version="1.0" encoding='ISO-8859-1'?>
        <data>
          <prueba>
            <carrito>
              <articulo>Gorra</articulo>
              <articulo>Camisa</articulo>
            </carrito>
          </prueba>
          <prueba>
            <carrito>
              <articulo>Jabon</articulo>
              <articulo>Detergente</articulo>
            </carrito>
          </prueba>
          <prueba>
            <carrito>
              <articulo>PS3</articulo>
              <articulo>Xbox</articulo>
            </carrito>
          </prueba>
          <prueba>
            <carrito>
              <articulo>Cds</articulo>
            </carrito>
          </prueba>
        </data>
```

A.1.1 XML Schema

Un esquema XSD, al igual que una DTD, define la estructura de uno o más documentos XML; esto es, qué elementos y atributos pueden contener, en qué orden deben estar, y cuál puede ser su contenido.

Un esquema es básicamente una colección de definiciones de tipos y declaraciones de elementos cuyos nombres pertenecen a un determinado espacio de nombres llamado espacio de nombres de destino. Los espacios de nombres de destino hacen posible la distinción entre definiciones y declaraciones de diferentes vocabularios.

En un principio existían dos sistemas fundamentales para describir la apariencia

de los documentos XML: las DTD y XML Schema. Este último es más complejo que otras alternativas anteriores, pero supuso un importante paso hacia adelante en la estandarización de XML. Nos centraremos en la explicación de los Esquemas XML, dado que el sistema desarrollado en este proyecto manipula este tipo de documentos.

En un esquema podemos dividir un documento en dos tipos de contenidos: simples y complejos. Los elementos del tipo simple son aquellos que sólo pueden contener texto. Los del tipo complejo pueden contener otros elementos, incluso atributos. Los atributos son considerados del grupo simple, ya que contiene solamente texto.

Hay cuatro clases fundamentales de tipos complejos:

- Elementos vacíos, que no contengan ni atributos, ni elementos ni texto.
- Elementos no vacíos, que contengan atributos, pero nunca elementos ni texto.
- Elementos que contengan otros elementos hijos o atributos, pero no texto.
- Elementos de "contenido mixto", que contengan una combinación de elementos hijos, atributos y/o texto, especialmente elementos y texto.

Todos pueden contener atributos. Tanto los tipos simples como los personalizados pueden tener nombre, en cuyo caso podrán utilizarse en otras partes del esquema, o bien ser anónimo, un cuyo caso sólo se utilizan en el elemento en el que aparece la definición.

En XML Schema, el contexto es muy importante. Los componentes del esquema, bien sean elementos, atributos, tipos simples o complejos, grupos o grupos de atributos, que son declarados en el nivel superior del esquema (debajo del elemento xsd:Schema*)*, son considerados declarados globalmente, y están disponibles para ser utilizados en el resto del esquema. Sin embargo, las declaraciones globales de elementos no determinan el lugar donde puede aparecer un elemento en el documento XML, sólo su apariencia. Por tanto, debemos hacer referencia a la declaración global de un elemento para que realmente aparezca en el documento XML. La única excepción a esta regla está destinada para el elemento raíz, el cual es referenciado automáticamente, independientemente de si se ha declarado globalmente o no.

Cuando se define un tipo complejo, éste puede hacer referencia a los elementos existentes declarados globalmente, o bien declarar y definir nuevos elementos. Estos elementos declarados localmente están limitados a la definición de tipo complejo en el cual han sido declarados y puede que no se utilicen en ninguna otra parte del esquema.

Además, sus nombres sólo necesitan ser únicos en el contexto en el que aparecen. Estos elementos son referenciados automáticamente, es decir, la posición en la que son definidos también determina en qué parte del documento XML deben aparecer.

Un esquema es un documento XML que sólo contiene texto y aparece con la extensión .XSD. Comienza con una declaración XML estándar, seguida de una declaración del espacio de nombre de XML esquema. En los documentos XML que se basen en ese esquema, incluiremos una referencia al archivo .XSD. Por ejemplo:

```
<?xml version=”1.0 “?>

<xsd:schema xmnls:xsd= http://www.w3.org/2001/XMLSchema>

    (conjunto de reglas del esquema)

</xsd:schema>
```

Los elementos utilizados en la creación de un esquema “proceden” del espacio de nombres: *http://www.w3.org/2001/XMLSchema*. El elemento *schema* es el elemento raíz del documento en el que se define el esquema.

Para definir un elemento simple, utilizamos la sintáxis:

```
<xsd:element name="xxx" type="yyy"/>
```

Los tipos de datos más utilizados son: *xsd:string*, *xsd:decimal*, *xsd:integer*, *xsd:boolean*, *xsd:date*, *xsd:time*.

Un elemento simple puede tener un valor por defecto y un valor “fijo”. Esto se indica mediante los atributos default y fixed.

```
<xsd:element name="color" type="xsd:string" default="red"/>
```

Los atributos se deben declarar de forma similar a los “elementos simples”. Si un elemento puede ir acompañado de atributos, el elemento se deberá declarar como un elemento “complejo”. Un atributo se declara de la siguiente forma:

```
<xsd:attribute name="xxx" type="yyy"/>
```

Los atributos tienen un tipo de dato: : *xsd:string*, *xsd:decimal*, *xsd:integer*, *xsd:boolean*, *xsd:date*, *xsd:time*. Por ejemplo:

```
<xsd:attribute name=“idioma" type="xs:string"/>
```

Los atributos pueden tener valores por defecto y valores fijos:

```
<xsd:attribute name=“idioma" type="xsd:string" default=“ES"/>
```

Por defecto, los atributos son opcionales. Para indicar que un atributo debe ser obligatorio, se debe añadir a su declaración en el esquema es atributo “use”.

```
<xsd:attribute name="lang" type="xsd:string" use="required"/>
```

El atributo use puede tomar el valor “optional” si el atributo no es obligatorio (opción por defecto).

Las facetas o restricciones permiten restringir el valor que se puede dar a un elemento o atributo XML. Mediante restricciones podemos indicar que un valor debe estar comprendido en un rango determinado, debe ser un valor de una lista de valores “cerrada”, o debe ser mayor o menor que otro valor…

Tipos de facetas:

- Valor comprendido en un rango.
- El valor está restringido a un conjunto de valores posibles.
- Restringir el valor de un elemento a una serie de caracteres.
- Longitud de los valores de los elementos.

Un ejemplo de una faceta que restringe un elemento a tres valores posibles sería el siguiente:

```
<xsd:element name="car">
        <xsd:simpleType>
                <xsd:restriction base="xsd:string">
                        <xsd:enumeration value="Audi"/>
                        <xsd:enumeration value="Golf"/>
                        <xsd:enumeration value="BMW"/>
                </xsd:restriction>
        </xsd:simpleType>
</xsd:element>
```

Para definir elementos complejos se utiliza la siguiente sintáxis:

```
<xsd:element name="employee">
        <xsd:complexType>
                <xsd:sequence>
                <xsd:element name="firstname" type="xsd:string"/>
                <xsd:element name="lastname" type="xsd:string"/>
                </xsd:sequence>
        </xsd:complexType>
</xsd:element>
```

El conjunto de reglas de un esquema podría ser por ejemplo:

```
<?xml version=”1.0 “?>
<xsd:schema xmnls:xsd= http://www.w3.org/2001/XMLSchema>

<xsd:annotation>
 <xsd:documentation xml:lang="es">
    Esquema de hoja de pedido
 </xsd:documentation>
</xsd:annotation>

<xsd:element name="hojaPedido" type="TipoHojaPedido"/>
<xsd:element name="comentario" type="xsd:string"/>

<xsd:complexType name="TipoHojaPedido">
 <xsd:sequence>
  <xsd:element name="enviarA" type="direccion"/>
  <xsd:element ref="comentario" minOccurs="0"/>
  <xsd:element name="elementos" type="Elementos"/>
```

```
 </xsd:sequence>
 <xsd:attribute name="fechaPedido" type="xsd:date"/>
</xsd:complexType>
<xsd:complexType name="direccion">
 <xsd:sequence>
  <xsd:element name="nombre" type="xsd:string"/>
  <xsd:element name="calle" type="xsd:string"/>
  <xsd:element name="ciudad" type="xsd:string"/>
 </xsd:sequence>
<xsd:attribute name="pais" type="xsd:NMTOKEN" />
</xsd:complexType>

<xsd:complexType name="Elementos">
 <xsd:sequence>
  <xsd:element name="elemento" minOccurs="0" maxOccurs="unbounded">
   <xsd:complexType>
    <xsd:sequence>
     <xsd:element name="nombreProducto" type="xsd:string"/>
     <xsd:element name="cantidad">
      <xsd:simpleType>
       <xsd:restriction base="xsd:positiveInteger">
        <xsd:maxExclusive value="100"/>
       </xsd:restriction>
      </xsd:simpleType>
     </xsd:element>
     <xsd:element name="precio" type="xsd:decimal"/>
     <xsd:element ref="comentario" minOccurs="0"/>
     <xsd:element name="fechaEnvio" type="xsd:date" minOccurs="0"/>
    </xsd:sequence>
    <xsd:attribute name="numProducto" type="SKU" use="required"/>
   </xsd:complexType>
  </xsd:element>
 </xsd:sequence>
</xsd:complexType>

<!-- Stock Keeping Unit [Código de Almacenaje], -->
<!-- un código para identificar productos -->

<xsd:simpleType name="SKU">
 <xsd:restriction base="xsd:string">
  <xsd:pattern value="\d{3}-[A-Z]{2}"/>
 </xsd:restriction>
</xsd:simpleType>
</xsd:schema>
```

El esquema está formado por un elemento *schema* y una variedad de subelementos, entre los que se destacan: *element* (elemento), *complexType* (tipo complejo), y *simpleType* (tipo simple) que determinan la aparición de elementos y su contenido en los documentos instancia (los documentos XML).

Cada uno de los elementos del esquema tiene el prefijo “xsd:” que está asociado con el espacio de nombres de XML Schema. El prefijo *xsd:* es usado por convención para denotar el espacio de nombres del XML Schema, aunque puede utilizarse cualquier prefijo. El mismo prefijo, y por tanto la misma asociación, también aparece en los nombres de los tipos simples predefinidos, por ejemplo *xsd:string*. El propósito de la asociación es identificar los elementos y tipos simples como pertenecientes al vocabulario del lenguaje XML Schema y no al vocabulario del autor del esquema.

A.1.2 XSLT

El XSLT forma parte de la familia de lenguajes del XSL (siglas de *Extensible Stylesheet Language),* basados en el estándar XML que permite describir cómo la información contenida en un documento XML cualquiera debe ser transformada o formateada para su presentación en un medio.

Desde 1997 varias empresas informáticas como Arbortext, Microsoft e Inso se pusieron a trabajar en una propuesta de XSL (antes llamado "xml-style") que presentaron al W3C y cuyo fin era permitir modificar el aspecto de un documento. Con las hojas de estilo ya se podían lograr algunas mejoras en el aspecto del documento, pero XSL permite otras muchas aplicaciones como múltiples columnas, orden de visualización de los datos de una tabla, múltiples tipos de letra con amplia variedad en los tamaños, etc.

Actualmente hay varias versiones del estándar XSLT: la versión 1.0, que es la que implementan la mayoría de los procesadores, y se denomina "recomendación", es decir, para el consorcio W3, lo equivalente a un estándar, y la versión 2.0 , que llegó a ser “recomendación” en 2007. Algunos procesadores, como el Saxon, implementan esta última versión. Hay algunas diferencias importantes: el tratamiento uniforme de los árboles, uso de múltiples documentos de salida, y funciones definidas por el usuario que se pueden definir en XSLT, y no sólo en Java u otro lenguaje, como sucedía en estándares anteriores.

La familia de lenguajes XSL está compuesta por tres tipos:

XSLT que permite convertir documentos XML de una sintaxis a otra (por ejemplo, de un XML a otro o a un documento HTML).

XSL-FO (lenguaje de hojas extensibles de formateo de objetos), que permite especificar el formato visual con el cual se quiere presentar un documento XML.

XPath, o XML *Path Language*, es una sintaxis (no basada en XML) para acceder o referirse a porciones de un documento XML.

Las hojas de estilo separan la información (almacenada en un documento XML) de su presentación, usando en cada caso las transformaciones que sean necesarias para que el contenido aparezca de la forma más adecuada, según el tipo de fichero de salida que se quiera generar. Se pueden usar diferentes hojas de estilo, o incluso la misma, para

presentar la información de diferentes maneras dependiendo de la información o de la composición de los nodos del XML de entrada.

Una hoja XSL puede actuar por sí sola como único elemento de transformación y encargarse de dar formato a todo el documento de salida, o adjuntarse a una estructura de hojas XSLT que pueden combinarse entre ellas y encargarse de tratar diferentes secciones del fichero XML, siendo referenciadas mediante elementos *xsl:include* y *xsl:import*.

Una estructura de hojas XSLT ha de tener una hoja de estilo principal *(principal stylesheet module*) que inicialice la transformación, y a su vez, incorpore el resto de plantillas XSL directa o indirectamente.

Se conoce como incorporación directa cuando la misma hoja de estilos o plantilla llama a otra mediante *xsl:include* o *xsl:import* sin que intervenga ninguna otra plantilla de por medio. La incorporación indirecta se da cuando una plantilla puede llamar a otra plantilla, que aunque no ha sido incluida previamente mediante *xsl:include* o *xsl:import*, se ha incluido por medio de una plantilla que sí ha sido incluida. Esto resulta útil ya que XSLT no nos permite incluir más de una vez la misma plantilla en una estructura XSLT, y con una buena distribución podemos trabajar con todas las plantillas.

Como se ha visto anteriormente, un documento XML se compone de información estructurada en nodos en forma de árbol. Si se tiene en cuenta que un XSLT ha de recorrer un documento XML la sintaxis básica para el desarrollo de XSLT, son recorridos, bucles, y condicionales que navegan por la estructura del XML.

La cabecera para todos los documentos será la misma:

```
<xsl:stylesheet version="1.0"
xmlns="http://www.w3.org/1999/xhtml”
xmlns:xsl="http://www.w3.org/1999/XSL/Transform">


<xsl:output method="xml" version="1.0"
encoding="utf-8"
media-type="text/html"
doctype-public="-//W3C//DTD XHTML 1.0 Strict//EN"
doctype-system=http://www.w3.org/TR/xhtml1/DTD/xhtml1-strict.dtd indent="yes"/>
```

Una hoja de estilos se inicia con *xsl* mediante la etiqueta *xsl:stylesheet* para todos los documentos que hagamos. Se indica la versión 1 de la hoja de estilo con el atributo *version* y el cumplimiento de las recomendaciones de la W3C junto con su *namespace* (criterio y restricciones de la recomendación sobre la estructura de tipos de elemento y nombres de atributo), con *xmlns* y *xmlns:xsl.*

Se debe incluir también la salida que resultará de la transformación con la etiqueta *output*. En la salida se indica el método de entrada con el atributo *method*, la versión del *xslt* y la codificación de caracteres que seguirá la plantilla mediante *version* y *encoding*

respectivamente. Para finalizar la cabecera se indica el medio de salida después de la transformación con *media-type* y la validación que seguirá con *doctype-public*.

La siguiente etiqueta también es obligatoria, ya que indica el nombre del *template* que se va a usar. Se puede hacer marcándolo con *match* y la secuencia de nodos a la que actuará, o bien dotándola de un nombre por el que luego podrá ser llamada con el atributo *name*.

<xsl:template match="XMLData/nodes">

<xsl:template name="nodes">

Estas últimas instrucciones se usarán para cerrar el marcado XSLT, también es obligatorio ya que si no nos daría un error al parsear.

</xsl:template>

</xsl:stylesheet>

A.2 Java

Java es un lenguaje de programación orientado a objetos y la primera plataforma informática desarrollada por Sun Microsystems a principios de los años 90. La intención de Sun era crear un lenguaje con una estructura y una sintaxis similar a C y C++, aunque con un modelo de objetos más simple y eliminando las herramientas de bajo nivel, que suelen inducir a errores.

Los pilares en los que se sustenta Java son cinco: la programación orientada a objetos, la posibilidad de ejecutar un mismo programa en diversos sistemas operativos, la inclusión por defecto de soporte para trabajo en red, la opción de ejecutar del código en sistemas remotos de manera segura y la facilidad de uso.

El corazón de la Plataforma Java es el concepto común de un procesador "virtual" que ejecuta programas escritos en el lenguaje de programación Java. En concreto, ejecuta el código resultante de la compilación del código fuente, conocido como *bytecode*. Este "procesador" es la máquina virtual de Java o JVM (Java Virtual Machine), que se encarga de traducir (interpretar o compilar al vuelo) el *bytecode* en instrucciones nativas de la plataforma destino. Esto permite que una misma aplicación Java pueda ser ejecutada en una gran variedad de sistemas con arquitecturas distintas, siempre que con una implementación adecuada de la JVM. Este hecho es lo que ha dado lugar a la famosa frase: "*write once, run anywhere*" (escribir una vez, ejecutar en cualquier parte). La condición es que no se utilicen llamadas nativas o funciones especificas de una plataforma y aun así no se asegura completamente que se cumpla una verdadera independencia de plataforma.

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 bibliotecas estándar, que contiene mucha de las funciones reutilizables disponibles en los sistemas operativos actuales.

La aplicación de Java es muy amplia. El lenguaje se utiliza en una gran variedad de dispositivos móviles, como teléfonos y pequeños electrodomésticos.

A.3 Oracle y PL/SQL embebido en C

El servidor de bases de datos Oracle permite almacenar y manipular datos de diferente índole (imágenes, sonidos, texto, caracteres, números, etc.). Hoy en día la última versión del servidor de datos es la 11. Oracle se compone de una serie de herramientas:

Un entorno de edición en línea que incorpora un intérprete de SQL, llamado SQL*PLUS.

Un lenguaje procedimental que permite utilizar estructuras de control y variables para elaborar programas que accedan a la base de datos donde se pueda utilizar comandos SQL, conocido como PL/SQL (Procedural Language for SQL). Este lenguaje es reconocido y procesado también por SQL*PLUS.

Una serie de bibliotecas para la programación utilizando otros lenguajes. Esta biblioteca conocida como OCI (Oracle Call Interfaces) fue la solución inicial al problema de desarrollar sistemas cliente/servidor. Hoy en día ORACLE provee una biblioteca propietaria de funciones para realizar comunicación con servidores de datos utilizando Java, la cual es conocida como JDBC (Java Database Connection).

Una serie de pre-procesadores (pre-compiladores) de SQL embebido, que constituyó la primera solución al problema de desarrollar programas para bases de datos. Existieron pre-compiladores que aceptaban instrucciones en un lenguaje de programación particular de tercera generación (en el caso de ORACLE los lenguajes ofrecidos era ADA, PL/I, COBOL, FORTRAN y C) junto con instrucciones del lenguaje SQL. Estas herramientas eran conocidas como Pro*ADA, Pro*PL/I, Pro*COBOL, Pro*Fortran y Pro*C.

En nuestro caso, se usará como lenguaje de programación C y se incluirán las instrucciones del lenguaje SQL para realizar las consultas necesarias a la base de datos.

El uso del lenguaje C en un entorno de desarrollo Linux es lo más habitual y práctico, ya que nos proporcionará una gran flexibilidad a la hora de programar. Utilizaremos todas las bibliotecas disponibles que sean de utilidad para el desarrollo de nuestro sistema. Será necesario el uso de bibliotecas para la sincronización entre procesos, todas

aquellas que tengan relación con semáforos, pipes, señales, sockets,…etc. Se incluirán librerías como <sys/types.h>, <sys/sem.h>, <semaphore.h>…etc.

A.4 Lenguajes de impresión

A.4.1 PCL

PCL (Printer Command Language) es un lenguaje de descripción de páginas desarrollado por Hewlett-Packard, a principios de los ochenta, para impresoras. No sólo es utilizado por impresoras HP, ya que muchos fabricantes de impresoras han adoptado al lenguaje como un estándar. Es más simple que PostScript y consume menos recursos.

Además del texto real que se imprime, el PCL consta en gran medida de comandos diseñados para accionar diversas características y capacidades de la impresora. Los comandos PCL tienen la forma de secuencias de escape, ya que son cadenas de caracteres que comienzan con un carácter de escape. Las versiones más nuevas de PCL tienen una secuencia de escape para iniciar el modo HP-GL, que permite la transmisión de gráficos vectorizados. Hay 6 clases de PCL que han ido apareciendo con nuevas funcionalidades conforme a las necesidades del nuevo hardware disponible.

Las instrucciones PCL se pueden dividir en cuatro grupos según el tipo:

Claves de control: consisten en un carácter ASCII no imprimible que representa función. Por ejemplo un retorno de carro provocará un salto de línea.

Comandos PCL: controlan todas las características de la impresora excepto aquellas relativas a los gráficos vectorizados.

Comandos HP-GL: son comandos formados por un código de dos letras seguido por una serie de parámetros que especifican cómo debe la impresora procesar el comando.

Comandos PJL: son comandos orientados a la configuración de las impresoras. Permiten que la impresora se comunique con el ordenador de manera bidireccional, intercambiar información del estado del trabajo e identificación de impresora y controlar el PDL que debe usar la impresora para un trabajo específico y otras funciones del panel de control de impresoras. Los comandos PJL están limitados al control de la impresora a nivel de trabajo y no están comprendidos en la impresión de documentos individuales.

Cada vez que se imprima un nuevo trabajo hay una serie de características de impresión que están establecidas por defecto de fábrica. Estas características varían en función del modelo de impresora y podrán ser modificadas por el usuario desde el panel de control (estas opciones se guardan aunque se apague la impresora) o bien mediante comandos antes de que se imprima el trabajo actual. Algunas de estas opciones de configuración

son: número de copias, selección de fuente, impresión a doble cara, número de bandeja, tamaño del papel, orientación,…etc.

Normalmente cuando se va a imprimir un trabajo primero se indican los comandos de control sobre todo el trabajo seguidos de los comandos de control de página y por último se indican los datos a imprimir.

Los comandos de control pueden establecer el número de copias, los márgenes de las páginas, el desplazamiento del cursor (horizontalmente y verticalmente) o incluso realizar un "reset" de la impresora.

Los comandos de control de página y de los datos se asocian a cada página del trabajo que se va a imprimir. Estos comandos determinan algunas características como tamaño de página, bandeja de donde obtener el papel, márgenes, orientación del papel, rotación del texto, posicionamiento del cursor y espacio del texto (espacio entre columnas y filas).

El posicionamiento del cursor puede ser absoluto, tomando como referencia la esquina superior izquierda del área de impresión, o relativo respecto a la posición actual del cursor. Las unidades de desplazamiento pueden ser unidades PCL (una unidad PCL es 1/72 pulgadas), decipuntos (es la décima parte de una unidad PCL) o columnas, en el eje X y filas, en el eje Y.

Las fuentes son un grupo de símbolos que tienen características similares y son descritas mediante una colección de símbolos disponibles, espacio entre caracteres (fijo o proporcional), número de caracteres por pulgada (sólo en fuentes con espacio fijo), altura de los caracteres, estilo (negrita, normal, itálica,..), densidad, tipo de letra, orientación,…etc. Hay una serie de fuentes que vienen dadas con la impresora y residen de forma permanente en la memoria ROM. Al igual que con otros lenguajes, se pueden descargar más fuentes.

En muchos de estos comandos PCL se utilizan la letra 'l' minúscula y el número '1' o la letra 'O' mayúscula y el número '0'. La figura A.1 muestra un comando de impresora típico (se trata de un comando que establece la orientación de la página).

Figura A.1 Figura de un comando PCL

Las secuencias de escape se pueden combinar en una sola cadena de secuencia de escape. Cuando se combinan varios códigos, hay que tener en cuenta tres reglas importantes:

1. Los dos primeros caracteres después del carácter E_C (los caracteres parametrizados y de grupo) deben ser los mismos en todos los comandos que se combinen.

2. Cuando se combinan varias secuencias de escape, se debe cambiar a minúscula el carácter en mayúscula (terminación) de cada secuencia de escape individual.

3. El carácter final de la secuencia de escape combinada debe ser una mayúscula.

La cadena de secuencias de escape que aparece a continuación se enviaría a la impresora para seleccionar papel Legal, con orientación horizontal y ocho líneas por pulgada:

E_C &l3A E_C &l1O E_C &l8D

La siguiente secuencia de escape envía los mismos comandos de impresora combinándolos en una secuencia más corta:

E_C &l3a1o8D

Los comandos PJL deben preceder a los comandos PCL, ya que se usan para gestionar la impresión. Un ejemplo de estos comandos puede ser la siguiente cabecera:

E_C %-12345X@PJL JOB NAME="Sin título - Bloc de notas"
@PJL COMMENT SET USERNAME="Administrador"
@PJL COMMENT SET MEDIATYPE=MEDTYPEPAPER
@PJL COMMENT SET DEVICE=PRINTER
@PJL COMMENT SET TONERREDUCTION=GENERICOFF
@PJL SET DUPLEX=OFF
@PJL COMMENT SET SORTERMODE=GENERICON
@PJL COMMENT SET BOOKLET=GENERICOFF
@PJL COMMENT SET STAPLE=GENERICOFF
@PJL SET RESOLUTION=600
@PJL ENTER LANGUAGE=PCL

Se puede apreciar que son comandos de configuración de la impresión como, por ejemplo, la resolución, impresión a doble cara, encuadernación, grapado,…etc. El último comando que aparece será necesario para dar paso a los comandos PCL, que ya hemos visto anteriormente.

A.4.2 ZPL

ZPL es uno de los lenguajes de programación propiedad de *Zebra Technologies Corporation* utilizado para la comunicación con las impresoras Zebra. Contiene una gran variedad de instrucciones que son enviadas a la impresora para permitir crear diferentes tipos de etiquetas. Las etiquetas pueden incluir texto, gráficos o códigos de barras. Las instrucciones de este lenguaje permiten posicionar cualquier elemento dentro de la etiqueta en la posición deseada. Las imágenes deben tener formato binario o hexadecimal para poder ser interpretadas.

Las instrucciones ZPL consisten en un código mnemotécnico de dos caracteres seguido de una cadena de parámetros. Su programación se basa en caracteres ASCII imprimibles, lo que permite que la transferencia de datos entre computadora e impresora sea más sencilla. No usa secuencias de escape o códigos de control. Existen unas pocas instrucciones que permiten modificar la configuración de la impresora. Fundamentalmente hay dos tipos de instrucciones: de formato y de control.

Las instrucciones de formato definen: etiquetas, campos de texto, campos alfanuméricos, campos con códigos de barras, gráficos, formatos por defecto, rotación de elementos...etc. Pueden tener diferentes parámetros como, por ejemplo, la longitud de la etiqueta, coordenadas del texto, tipo de campo, texto del campo y otras cosas. La mayoría de las instrucciones de formato son independientes del orden de ejecución. Otras en cambio, dependiendo del método de procesamiento utilizado, se deben anteponer unas a otras. Van precedidas por el símbolo '^', que en hexadecimal es '5E'. Algunas de estas instrucciones son:

^LH (Label Home): establece es extremo superior del área imprimible de la etiqueta.

^LL (Label Length): establece la longitud de la etiqueta.

^LR (Label Reverse): invierte toda la etiqueta 180º.

^LS (Label Shift): realiza un desplazamiento de la etiqueta.

^JM (Set Dots/Millimeter): establece el número de puntos por milímetro, que dependerá del tipo de cabezal de la impresora.

^PO (Print Orientation): cambia la orientación de la etiqueta en 90º, 180º o 270º.

^PF (Slew Dot Rows): desplaza la cabeza de impresión tantas filas de puntos como se le indique para continuar la impresión a partir de ese punto.

Las instrucciones de control suelen ser precedidas por el carácter:' ~ ', que en hexadecimal es '7E'. Cuando se especifica una de estas instrucciones, se aplica inmediatamente sobre la impresora. Puede tratarse de borrar la memoria o dar una etiqueta en blanco. Pueden interrumpir cualquier instrucción de formato que se

encuentre esperando en el buffer de la impresora. Este tipo de instrucciones también se usa para configurar la impresora.

En una instrucción no es necesario definir siempre todos los parámetros. Se puede cambiar algún parámetro y dejar el resto con el valor por defecto poniendo el comando seguido de tantas comas como parámetros precedan al que queramos cambiar, como por ejemplo: ^AA, ,60 (el resto de parámetros por detrás no hace falta contarlos).

Las instrucciones pueden hacer referencia a un nombre de dispositivo refiriéndose a una de sus áreas de almacenamiento. Esto permite el almacenamiento, la extracción, copia o borrado de los objetos ZPL desde y hacia dichas áreas. Para ello se hace corresponder una letra a cada una de estas áreas:

R: se refiere a la memoria DRAM.

B: se refiere a la memoria opcional (una tarjeta o memoria adicional de fábrica).

E: se refiere a la memoria extra EPROM de sólo lectura.

Z: se refiere a la memoria interna ZPL

Las impresoras Zebra suelen tener 8 fuentes bitmap y 1 escalable. Suele haber fuentes disponibles para descargar. El tamaño del carácter y la densidad suelen depender del número de puntos por mm y del tipo de papel usado. Hay 3 densidades: 6 puntos/mm, 8 puntos/mm y 12 puntos/mm. A partir del tamaño establecido por defecto para las fuentes internas de bitmap, se pueden incrementar los tamaños de los caracteres en factores de entre 2 y 10 y así, obtener diferentes tamaños (los incrementos a lo ancho y a lo alto se pueden modificar de forma independiente). La fórmula para el incremento a lo alto sería la siguiente:

Altura de base x Factor de magnificación = Valor del parámetro de altura

Se aplicaría el mismo principio para el incremento a lo ancho. Por ejemplo, si a una fuente cuyo tamaño normal es 9 puntos de alto y 5 puntos de ancho se le aplica un factor de magnificación igual a 3, se obtendrá un carácter de 27 puntos de alto y 15 de ancho.

Los espacios entre caracteres pueden ser fijos o proporcionales. Para los tipos de fuente de la ‘A’ a la ‘H’ serán fijos y por lo tanto, será siempre igual a un número de puntos.

La línea de base es una línea imaginaria a partir de la cual descansa la parte más baja de los caracteres que no son descendentes. Para aquellos caracteres descendentes se utilizará el espacio que quede entre la línea de base y la zona inferior que delimita la celda del carácter. La línea de base puede ser un número fijo establecido o situarse un número de puntos (¾ multiplicado por tamaño de la celda) por debajo de la zona superior que delimita la celda del carácter.

Las impresoras Zebra imprimen una gran variedad de códigos de barras. Los códigos de barras suelen tener como parámetros un carácter de comienzo, dígito de checksum, carácter de fin, zona vacía y el campo con los datos. No todos los códigos de barras requerirán todos estos elementos. Las instrucciones para imprimirlos empiezan con '^B' seguido de un número o letra que indique el tipo de código.

Se pueden destacar otras características de programación de la impresora como el hecho de poder imprimir lotes de etiquetas estableciendo tiempos de pausa entre diferentes bloques de impresión. También es posible descargar nuevas imágenes, formatos o fuentes y guardarlos en la memoria interna de la impresora.

A.4.3 PGL

PGL es uno de los lenguajes de programación propiedad de *Printronix*, utilizado para la comunicación con las impresoras Printronix. Contiene una gran variedad de instrucciones que son enviadas a la impresora para permitir crear diferentes tipos de etiquetases un lenguaje que ofrece *forms on-line*, códigos de barras y capacidades para generación de texto y gráficos. Es compatible con versiones anteriores del lenguaje de programación IGP (*Intelligent Graphics Printing*). Las instrucciones de este lenguaje permiten posicionar cualquier elemento dentro de la etiqueta en la posición deseada. Las imágenes permiten varios formatos para poder ser interpretadas: *PCX*, *TIFF* y hexadecimal.

Las impresoras que soportan este lenguaje tienen varios modos de funcionamiento y esto condiciona las instrucciones que pueden ejecutar. En modo de *pausa* ignora cualquier comando que reciba (excepto el comando *LISTEN*). En modo *normal* puede recibir e imprimir documentos y en modo de *creación* puede recibir las instrucciones PGL y los datos que va a imprimir y almacenarlos en memoria. El modo *ejecución* se pueden imprimir los documentos almacenados durante el modo de *creación*. En modo de *configuración* se pueden enviar comandos a la impresora para fijar algunas opciones de impresión que afectarán a los siguientes documentos que se impriman. Este lenguaje de impresión tiene la peculiaridad de que permite imprimir más de un documento manteniendo unos datos fijos y otros que varían de un documento a otro.

Las instrucciones PGL son palabras que deben escribirse en mayúsculas. Cada palabra representa un comando diferente y se acompaña de una serie de parámetros separados por ';'. Los comandos deben ir seguidos de un salto de línea y el final de sus parámetros se detecta al encontrar la palabra 'STOP'. Las instrucciones de control son las que permiten el cambio de modo de la impresora y van precedidas por el símbolo '~' (126 en ASCII). Cada uno de sus modos tiene una lista de instrucciones que son las que permite ejecutar mientras dure ese modo. A continuación se presenta un ejemplo con diferentes modos que imprime un conjunto de caracteres.

```
~CREATE;TEST;288          (Se inicia el modo de creación 'Create', introduciendo un nombre
                            de form)
VDUP;3;6                  (Se duplica la cadena de texo del comando Alpha)
ALPHA                     (Comando Alpha)
I;6;5;4;4;-00001;*12345*  (Cadena de parámetros)
STOP                      (Fin del comando Alpha)
VDUP;OFF
END                       (Fin del modo de creación 'Create')
~EXECUTE;TEST             (Se imprime el form)
~NORMAL                   (Se vuelve al modo normal)
```

Al ser un lenguaje de impresión para etiquetadoras es similar en algunos aspectos a ZPL, pero ofrece mayor diversidad de comandos. Permite imprimir un mayor número de códigos de barras y ofrece un mayor número de opciones para su impresión. Al igual que las impresoras *Zebra* permite la impresión por transferencia termal o de forma directa y mediante papel continuo o discontinuo, pudiendo cortar las etiquetas total o parcialmente. La unidad de medida que usa para posicionar los elementos en la etiqueta son las pulgadas. Para ver el conjunto de caracteres y de fuentes que soporta se deben consultar las especificaciones de la impresora en la que se desea imprimir y en caso de que se necesite alguna fuente, permite almacenar nuevas fuentes en memoria de forma permanente.

A.5 Sincronización entre procesos

El producto SILO se compone de una serie de módulos bien diferenciados. Entre ellos, hay uno que presta servicios *Core* a más bajo nivel y es en el que se incluirá nuestro sistema como una parte más de él.

Hay una serie de procesos asociados a cada tabla de datos de tipo "TL_NOMBRETABLA". Esta tabla recibe un mensaje, el tipo de mensaje, de dónde procede y a quién va dirigido. Cada tabla de este tipo puede tener asociadas de 0 a N tablas. Estos procesos funcionan con dos tipos de flags: flag de proceso y flag de grupo. Por tanto, todo proceso pertenece a un grupo de procesos. Los flags se utilizan para modificar el comportamiento del proceso. El comportamiento normal de estos procesos es el siguiente: el proceso ejecuta un bucle de modo que permanece dormido y se despierta cada cierto tiempo (unos 30 segundos) para comprobar el estado de las tablas y ver si alguien ha escrito en ellas. Si las tablas no se han modificado entonces se volverá a dormir. Si detecta que otro proceso escribe en sus tablas se despertará y extraerá la información que haya en ellas. Nuestro sistema se encargará de gestionar la información extraída. El flag de proceso indica el estado del proceso, de modo que si su valor es 1, el proceso estará despierto y si su valor es 0, estará dormido. El valor del flag de grupo es el que determina si el proceso está activo o no. Cuando se le manda a un proceso un flag de grupo con valor 1 el proceso desaparecerá de la lista de procesos activos. Para activarlo de nuevo se le deberá enviar un flag de grupo con valor 0.

Anexo B

Análisis del sistema

Este anexo presenta en la sección B.1 todos los requisitos que se consideraron durante la fase de análisis. La sección B.2 describe los casos de uso del sistema, que complementa el análisis del comportamiento del sistema.

B.1 Análisis de requisitos

B.1.1 Requisitos funcionales

Requisitos Seguridad

R1: Al iniciar la aplicación se solicitará un nombre de usuario y contraseña, de modo que cada usuario sólo tenga acceso a la información creada por él mismo.

Requisitos de Datos

R1: Los datos introducidos sobre cada opción deben ser del tipo y la forma especificados.

R2: Los datos deberán ser completos a ser posible, pero podrán tener campos vacíos si así se indica en otro campo con el que tenga relación.

R3: El identificador de cada dispositivo, cada input, cada opción seleccionada y cada output generado es único.

R4: Se almacenarán en la base de datos los esquemas XSD y los distintos documentos XML que sean necesarios, que contendrán información acerca de los datos de entrada, los datos de formato de las plantillas y los dispositivos y capacidades de los mismos.

R5: Se usarán plantillas XLT para almacenar la estructura de los documentos que se van a generar. A partir de estas plantillas y junto con los datos que deben contener, se generarán los documentos finales.

R6: Se almacenarán también plantillas de estilo XSL necesarias para generar todos los tipos de outputs posibles en el código nativo del dispositivo.

Requisitos de Operaciones disponibles por la aplicación

R1: Operaciones de impresión y envío de distintos tipos de inputs reconocidos por la aplicación a los diferentes dispositivos almacenados en la base de datos. Permitirá la impresión o envío de cualquier tipo de output que se suela crear dentro de la aplicación SILO.

R2: Operaciones de modificación de layout de ciertos inputs, permitiendo crear nuevas plantillas con un nuevo diseño que modifique la colocación de sus elementos. Permitirá guardar las nuevas plantillas de rediseño de un tipo de output para el usuario que lo solicite.

R3: Operaciones de creación de nuevas plantillas con diferentes opciones de impresión para un usuario que lo solicite.

R4: Operaciones de impresión y envío de inputs a los diferentes dispositivos mediante comunicación directa con el usuario o mediante impresión directa sin necesidad de pedir datos al usuario.

R5: Operaciones de reconocimiento de las capacidades de un dispositivo, examinando si la impresión es factible con las opciones de impresión seleccionadas. En caso de no ser posible, realiza búsqueda de equivalencias para poder imprimir el input con las características disponibles en el dispositivo elegido.

R6: Operaciones de transformación de inputs mediante la selección de plantillas adecuadas para convertirlas al formato de salida esperado y traducido al lenguaje de comandos nativo del dispositivo destino.

Requisitos de Navegación

R1: Las ventanas de la aplicación serán gestionadas por teclado y ratón.

R2: Las ventanas de la aplicación que formarán parte de la misma se encargarán por un lado, de gestionar la configuración y la instalación del driver y por otro lado, de la impresión o envío de documentos a partir de sus plantillas, que habrán sido creadas previamente.

R3: Antes de empezar a usar la aplicación el usuario deberá realizar la instalación de la misma mediante un archivo de "setup".

R4: La comunicación entre las pantallas para la instalación del driver será en ambas direcciones, por medio de opciones de "Atrás" y "Siguiente" o "Aceptar". Durante el proceso de instalación se pedirá al usuario la aceptación de las licencias de uso correspondientes y se seleccionará un directorio destino para almacenar los archivos necesarios que permitan su funcionamiento.

R5: Se deberá informar al usuario de cualquier error que pudiera ocurrir durante la instalación y si no lo hubiera, finalizará informando al usuario de su correcta instalación.

R6: Al usar la aplicación, la primera pantalla solicitará un nombre de usuario y su correspondiente contraseña. De este modo se realiza la gestión de las plantillas de cada usuario por separado. Se puede tener un nombre de usuario común que represente a un grupo de usuarios que vayan a compartir las mismas plantillas para generar sus documentos.

R7: La última pantalla será una simple pantalla de finalización de aplicación indicada con la opción de "Finalizar", una vez que se nos indique que el documento se ha imprimido o enviado correctamente.

R8: Se podrán crear nuevas plantillas con las opciones seleccionadas para la impresión cada vez que se utilice la aplicación. Estas plantillas se guardarán y serán mostradas en pantalla para permitir que el usuario las utilice cuando lo desee.

R9: Al seleccionar las opciones de impresión del dispositivo aparecerán marcadas las opciones por defecto del fabricante la primera vez que se utilice la aplicación.

R10: En las pantallas para la impresión de documentos se mostrarán diferentes opciones para la creación o edición de plantillas o bien para seleccionar una plantilla de usuario creada previamente. Una vez que se haya elegido una plantilla para el documento que se desea imprimir, se deberá elegir primero el dispositivo de impresión y a partir de ese punto, se mostrará la opción de imprimir el documento con la palabra "Imprimir".

Requisitos de Interfaz

R1: La interfaz gráfica de la aplicación se podrá manejar por ratón y teclado para introducir datos a la aplicación, pero no siempre requerirá datos del usuario.

R2: La aplicación usará pantallas emergentes para la comunicación de errores al usuario.

R3: Un error en la inserción de datos por parte del usuario, activará unas ventanas emergentes con un "Mensaje de error", que dependiendo del momento en el que tiene lugar el error mostrará un mensaje u otro.

R4: La interfaz gráfica permitirá seleccionar los dispositivos introducidos por el usuario en trabajos de anteriores o los dispositivos conectados en red, según sea el acceso local o remoto. Se mostrarán mediante una lista desplegable. Para que puedan ser utilizados la información acerca de sus capacidades deberá estar almacenada en la base de datos.

R5: Desde la interfaz gráfica se permitirá cerrar la aplicación.

R6: Las pantallas que componen la aplicación permiten la identificación del usuario, ver o cambiar las opciones de configuración, ver las opciones de plantillas que permiten su creación, edición o eliminación, ver o cambiar algunas opciones de impresión y seleccionar un archivo con los datos que se van a incluir en la plantilla seleccionada para su envío o impresión.

R7: La pantalla "RegistroUsuario" permite la identificación del usuario para entrar en la aplicación.

R8: La pantalla "OpcInicio" aparece una vez que el usuario se ha identificado y ofrece las opciones mínimas necesarias para la impresión del documento. Presentará una lista con las plantillas del usuario y la opción de introducir un archivo XML o de texto plano, que contenga los datos que van a complementar la plantilla elegida. Tiene una serie de botones suya selección permite las siguientes acciones:

- El botón "Imprimir" permite la impresión de un documento. Para ello antes deberá seleccionar un archivo que proporcione los datos de entrada y una plantilla para generar el documento. con las opciones de impresión actuales (si no se ha seleccionado ninguna se aplican las opciones por defecto).

- El botón “Opciones de configuración” permite al usuario, siempre que disponga de los permisos necesarios, visualizar las opciones de configuración y realizar los cambios que desee en el dispositivo que elija.
- El botón “Opciones de plantilla” permite continuar con la navegación por una serie de pantallas que sirven para la creación, edición, duplicación, eliminación o visualización de plantillas.

R9: La pantalla “OpcionesPlantilla” presenta una lista con las plantillas de usuario y distintos botones para la creación, edición o eliminación de plantillas. Presenta un botón “Atrás”, que permite retroceder a la pantalla “OpcInicio” para volver a la pantalla anterior.

R10: La opción de creación de plantillas permite seleccionar todas las opciones necesarias para definir una plantilla. Entre estas opciones estarán: el nombre de la plantilla, el tipo de documento que se va a generar, el dispositivo destino, el tamaño de la plantilla y los elementos que va a contener en cada página, así como su disposición en la misma.

R11: La opción de edición de plantillas permite ver las opciones seleccionadas para una plantilla, habiéndola elegido previamente. El usuario podrá modificar las opciones que desee.

R12: Al terminar de crear o editar una plantilla, se guardarán los datos de la plantilla y se volverá a la pantalla inicial. Si se ha creado una plantilla se mostrará la lista de plantillas del usuario, que incluirá la nueva plantilla creada. En caso de haber editado una plantilla, la lista permanecerá con las mismas plantillas, pero los cambios de la plantilla habrán sido guardados.

R13: Cuando se haya seleccionado la plantilla y el archivo que contenga los datos del documento a imprimir, aparecerá una pantalla que mostrará un resumen con el nombre del usuario, del archivo de datos, de la plantilla y del dispositivo destino. Si los datos son correctos se procederá a la impresión. Se dará la opción de volver a la pantalla anterior si se desea realizar algún cambio.

R14: Durante la impresión del documento se mostrará el estado de impresión, para ver cómo va progresando.

R15: Al finalizar la impresión, la pantalla “OpcSalir” mostrará un mensaje informativo que nos indique si la impresión ha sido correcta o no. Se ofrecerá la posibilidad de imprimir más documentos o salir de la aplicación.

B.1.2 Requisitos no funcionales

Requisitos de Capacidad de almacenamiento

R1: El sistema permitirá que se almacenen tantas plantillas e información correspondiente a los documentos, como sea posible en la base de datos, mientras no afecte a la eficiencia del sistema.

Requisitos de Recursos

R1: Deben ser necesarias las licencias de ejecución.

R2: La aplicación necesita un motor de base de datos, existente en el producto SILO de HP. El sistema de gestión de la base de datos será Oracle, a partir de la versión 10.

R3: Se necesita conexión a internet para la conexión a la base de datos por parte de la aplicación.

R4: Resultará imprescindible adquirir los distintos módulos que componen el producto SILO, ya que se aprovecharán los recursos necesarios.

R5: El sistema operativo sobre el que se desarrollará la aplicación será Linux Red Hat, en cualquiera de sus versiones 5 ó 6.

R6: La parte "front-end" de la aplicación requiere tener instalada la máquina virtual Java a partir de la versión 6.1.3.

R7: La aplicación se desarrollará en lenguaje C, utilizando las librerías que se consideren necesarias y las consultas a la base de datos se embeberán en el código C mediante el uso de PL/SQL.

R8: La comunicación entre los clientes, que realizan las peticiones de impresión, y el servidor se realizará mediante sockets.

R9: Se utilizarán parsers XML para analizar los documentos con dicho formato.

R10: Se necesitarán motores de transformación XSL (XSLT) para convertir los documentos de entrada con formato XML en el formato de salida deseado.

Requisitos de Interfaz

R1: La aplicación dispondrá de un conjunto de pantallas en formato gráfico.

R2: La aplicación se presentará en español al inicio de la instalación del driver.

R3: Los colores y secciones que conforman la interfaz cumplirán los estándares de accesibilidad y usabilidad.

Requisitos de Verificación

R1: Se comprobará que el listado de requisitos se verifica revisándolos directamente uno por uno una vez que la aplicación esté desarrollada.

Requisitos de Pruebas de aceptación

R1: El sistema será sometido a una serie de pruebas, en las que comprobará el correcto funcionamiento respecto a los requisitos especificados por el desarrollador del sistema.

Requisitos de Documentación

R1: Documento de análisis, es el documento que presentará la primera fase del sistema en el que se le expone textual y gráficamente los requisitos del sistema.

R2: Documentos de diseño, es el documento que presentará la segunda fase de desarrollo del sistema en el que podrá visualizar el formato de la interfaz gráfica a desarrollar. Además presentará el diseño de objetos mediante un modelado estático y

dinámico, así como el diseño del sistema a través de la arquitectura y su descomposición en módulos por sus distintas funcionalidades.

R3: Manual de usuario, es el documento que especifica la forma de uso de la aplicación.

R4: Manual de instalación, es el documento que especifica como instalar la aplicación en la maquina usuario.

Requisitos de Portabilidad

R1: La aplicación puede ser ejecutada en cualquier Sistema Operativo siempre y cuando contenga la máquina virtual de Java y tenga acceso a Internet.

Requisitos de Calidad

R1: Ley de protección de datos.

R2: La aplicación cumplirá una calidad mínima, siendo revisada durante el proceso de desarrollo.

R3: La aplicación tiene la propiedad interoperabilidad con el sistema gestor de la base de datos.

R4: La aplicación es tolerante a fallos previstos que puedan ocurrir.

R5: La aplicación es capaz de recuperarse tras la aparición de un fallo, y hacerlo con un costo mínimo, de tal manera que no modifica en exceso el rendimiento óptimo del sistema para continuar con su funcionalidad.

R6: La aplicación no presenta unas grandes capacidades de usabilidad, en cuanto aprendizaje y comprensión de uso del mismo por parte de un usuario no técnico, que no tenga ciertos conocimientos informáticos.

Requisitos de Fiabilidad

R1: El sistema será robusto a errores introducidos por el usuario.

R2: Todas los errores de la base de datos serán controladas.

R3: En caso de error se informará al usuario a través de la interfaz.

Requisitos de Mantenibilidad

R1: Como la aplicación no ofrece la posibilidad de modificación, actualización en la base de datos, estas operaciones serán realizadas por un administrador externo al sistema.

Requisitos de Salvaguarda

R1: El sistema ante un error que provoque la caída del mismo, asegura que no se perderán los datos de la aplicación.

B.2 Casos de uso

En esta parte del documento se presentan los casos de uso que se han identificado junto con los actores con los que se relaciona nuestra aplicación. Estos actores serán dos: un usuario que utilice la aplicación desde la parte de *front-end* o bien otra aplicación del entorno donde se encuentre nuestra aplicación. Este hecho hace diferenciar claramente dos comportamientos de nuestra aplicación: uno en el que gestiona los *inputs* de forma *desatendida* y los envía a otro sistema o dispositivo y otro en el que realiza sus gestiones de forma *atendida* por parte del usuario, introduciendo una serie de parámetros que se le indican. Se presentan a continuación todos los casos de uso posibles en el siguiente diagrama de casos de uso de la figura B.

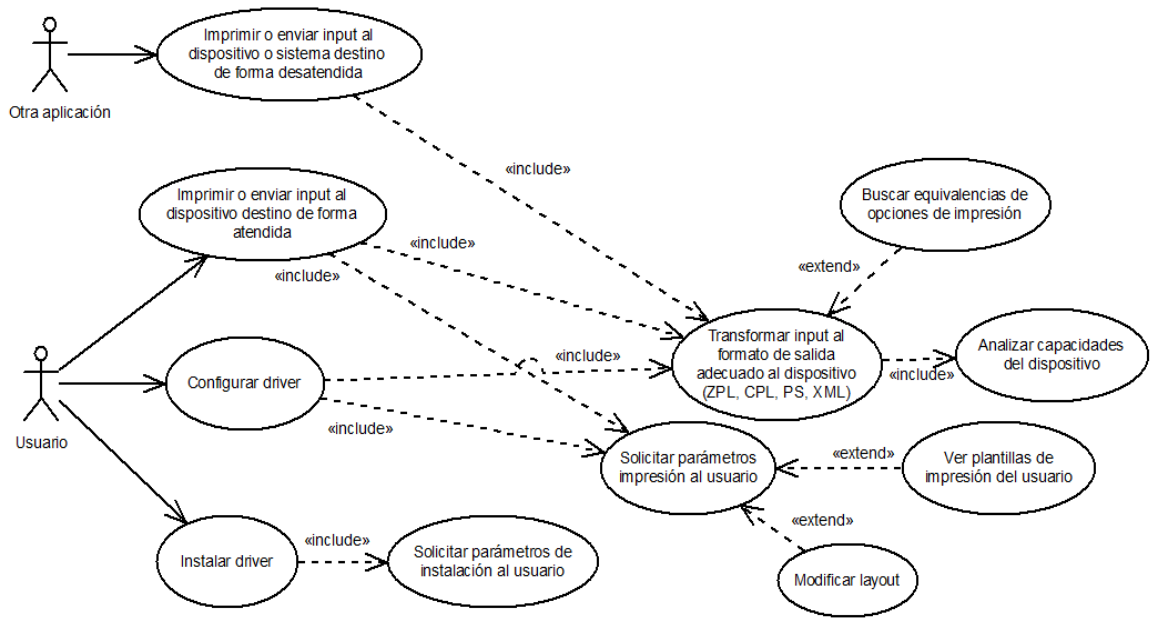

Figura B.2 Figura que representa los casos de uso de la aplicación

Primero presentaremos los casos que se relacionan con el actor "Usuario".

Caso de uso: Imprimir o enviar input a dispositivo destino de forma atendida

Actores: Usuario

Descripción: Imprimir o enviar un input solicitando primero parámetros de impresión o datos del envío al usuario. Esta acción puede incluir o no la selección de alguna de las plantillas predefinidas por el usuario que solicita la impresión. Para ello el sistema le pedirá que se identifique.

Curso normal de eventos:

1. El sistema muestra la pantalla principal y solicita la identificación del usuario. El sistema deberá validar los datos y si son correctos se le mostrarán sus plantillas de impresión. En caso de no ser válidos los datos introducidos, se mostrará un error de identificación y no aparecerán las plantillas de usuario.
2. El sistema muestra la pantalla de inicio que contiene una lista de sus plantillas y permite la impresión o envío de un input a un dispositivo o sistema destino.
3. El usuario selecciona una de sus plantillas y se muestra el dispositivo asignado a la plantilla.
4. El sistema comprueba la conexión y disponibilidad del dispositivo y si la comprobación es correcta, carga las opciones disponibles para ese dispositivo o sistema.
5. Se le proponen una serie de opciones al usuario: imprimir o enviar el input al destino directamente con las opciones por defecto del sistema, ver las opciones de impresión o envío, ver las opciones de plantillas, ver las opciones de configuración o salir de la aplicación cancelando la impresión o el envío.
6. El usuario tiene varias alternativas a elegir:
   6.a. El usuario elige modificar el layout del input y el sistema le muestra un editor de layout donde puede seleccionar elementos de las distintas páginas que se van a imprimir o enviar y modificar su posición actual por la que desee. Cuanto termine con el nuevo diseño podrá guardarlo si lo desea y volver a la pantalla anterior del punto 5.
   6.b. El usuario pulsa el botón adecuado para ver las opciones de impresión o envío y sigue a partir del punto 7.
   6.c. El usuario selecciona la opción de ver las opciones de configuración.
   6.d. El usuario sale del sistema cerrando el diálogo de ventanas para la impresión o envío del input actual.
   6.e. El usuario rellena los datos para la impresión de documentos y pulsa el botón “Imprimir”.
7. El usuario elige la alternativa 6.b., ver las opciones de impresión o envío del dispositivo seleccionado.
8. El sistema comprueba las capacidades del dispositivo o sistema seleccionado por el usuario y en base a ello mostrará las opciones de impresión o envío.
9. El usuario interactúa con el sistema mediante teclado y ratón, introduciendo los valores adecuados para cada opción seleccionada y navegando a través de diferentes pestañas que agrupan diferentes opciones relacionadas entre sí.
10. El usuario, una vez que ha seleccionado todas las opciones que desea para la impresión o envío del input, pulsa el botón “Aceptar” en cualquiera de las pestañas de opciones que se encuentre.
11. Se vuelve al punto 5.
12. El usuario elige la alternativa 6.e. yselecciona imprimir o enviar el input al dispositivo o sistema destino elegido previamente con las opciones seleccionadas en los pasos anteriores. Antes deberá seleccionar el número de

copias que se van a imprimir o enviar y entonces pulsará el botón y pasará al punto. El usuario esperará la impresión o envío del input.

13. El sistema transformará el input al formato de salida adecuado, que dependerá del dispositivo o sistema destino. Los formatos de salida posibles serán: ZPL, PCL y PGL.
14. El sistema envía el output correspondiente a la transformación del input al destino y muestra su estado de impresión o envío al usuario.
15. Cuando el output ha sido completamente enviado o impreso por el dispositivo o sistema destino se le comunica al usuario y se le muestra una pantalla de finalización.

Caso de uso: Configurar driver

Actores: Usuario

Descripción: Debe buscar la plantilla correspondiente para la configuración del driver y hacer una copia de ella para introducir la nueva configuración al sistema.

Curso normal de eventos:

1. El sistema busca la plantilla de configuración actual del dispositivo a configurar y realiza una copia.
2. El usuario puede introducir nuevos parámetros de configuración.
3. El sistema deberá introducir los cambios de configuración en la nueva plantilla y conectar con el dispositivo destino.
4. Se le ordena al dispositivo destino que imprima una hoja de prueba para verificar la nueva configuración.
5. Si la impresión es correcta y no se detectan errores se da por válida la nueva configuración y se guardan los cambios.

Caso de uso: Instalar driver

Actores: Usuario

Descripción: El usuario realiza la instalación del driver para poder usarlo posteriormente.

Curso normal de eventos:

1. El sistema muestra la pantalla principal de bienvenida al usuario con una breve información acerca del producto.
2. El usuario comenzará la instalación a partir de esa primera pantalla de bienvenida.

3. El sistema le muestra los términos de la licencia de uso y espera la conformidad del usuario, que deberá aceptarlos.
4. El usuario acepta los términos de la licencia de uso y empieza la instalación.
5. El usuario selecciona una carpeta para la instalación de los archivos necesarios para su ejecución y el sistema los va descargando en dicha carpeta.
6. Finalizará la instalación de todos los archivos necesarios para el funcionamiento correcto del sistema y mostrará una pantalla de fin de instalación al usuario para que cierre la aplicación.

Por último presentamos el caso de uso que se relaciona con el actor "Otra aplicación".

Caso de uso: Imprimir o enviar input a dispositivo destino de forma desatendida

Actores: Otra aplicación

Descripción: Imprimir o enviar un input sin solicitar parámetros de impresión o datos del envío al usuario. Deberá identificar el tipo de input que recibe y una vez identificado, gestionará su impresión o envío a un dispositivo destino sin que el usuario intervenga.

Curso normal de eventos:

1. El sistema recibe un input de entrada en forma de telegrama y deberá consultar la información de la tabla que almacena estos telegramas en la base de datos.
2. Una vez identificada y almacenada la información correspondiente, el sistema analiza los datos de entrada y genera un archivo XML con todos los datos de entrada recibidos.
3. El sistema analiza las capacidades de impresión o envío del dispositivo destino conforme a los datos recibidos. Para ello debe buscar las plantillas correspondientes al tipo de output que debe generar y al dispositivo destino.
4. En el caso de que no sean suficientes las capacidades del dispositivo destino para llevar a cabo la impresión o envío de los datos recibidos, deberá buscar las equivalencias necesarias para su impresión o envío.
5. Una vez que el sistema da por válida la impresión o envío al dispositivo, busca las plantillas necesarias para la generación del output correspondiente.
6. El sistema transformará el input al formato de salida adecuado, que dependerá del dispositivo o sistema destino. Los formatos de salida posibles serán: ZPL, PCL y PGL.
7. El sistema envía el output correspondiente a la transformación del input al destino y muestra su estado de impresión o envío al usuario.
8. Cuando el output ha sido completamente enviado o impreso por el dispositivo o sistema destino se le comunica al sistema finaliza el proceso de impresión existente.

Anexo C

Diseño del sistema

En este anexo se explican con mayor detalle los componentes internos del sistema.

C.1. Diseño de estructuras

En esta sección se van a detallar los componentes del sistema mediante su descomposición en los módulos que ya se describieron en el capítulo 3. Se detallan sus estructuras de datos para completar su descripción y comprender mejor el diseño realizado. En el caso del módulo *interfaz* de la aplicación de *front-end* se describen las clases que lo forman.

Primero se describen las estructuras correspondientes a los cuatro módulos de la aplicación *motor* y por último la *interfaz* perteneciente al *front-end*.

C.1.1 Módulo gestor de la base de datos

Este módulo tiene como operaciones básicas la conexión y desconexión de la base de datos, la consulta de información y el almacenamiento de los datos necesarios. Para la consulta y almacenamiento de datos comparte una serie de estructuras con el resto de módulos. Estas estructuras de datos tienen un diseño similar al de las tablas de la base de datos, ya que generalmente las consultas realizadas solicitan todos los datos almacenados en una misma tabla. Su diseño comprende un conjunto de estructuras que almacenan los datos relativos a las plantillas y sus elementos y otra estructura que contiene la información general de un documento, cuando se solicita su impresión. Dentro del conjunto de estructuras que contienen los datos de las plantillas se pueden diferenciar dos grupos, que vienen determinados por el tipo y estado de la petición y por el módulo con el que debe compartir la información. Un grupo se usa en el flujo de datos establecido con el *gestor de entradas* y el otro grupo con el *motor de impresión*. Esta diferencia se establece porque el *gestor de entradas* se comunica con la *interfaz* y de ella se recibe toda la información necesaria para la generación de plantillas, que será equivalente a la que se almacene en las tablas de la base de datos. En cambio, en la impresión de documentos, el *motor de impresión* no necesita toda esa información y sólo se le comunica parte, por lo que las estructuras no son las mismas.

Primero se detallan las estructuras comunes, que usan todos los módulos de la aplicación *motor*, tanto para la gestión de plantillas, como para la impresión de documentos. Sus TAD son los siguientes:

- TAD info_plantilla: contiene los datos generales de la plantilla procedentes de la información almacenada en tabla *T_PLANTILLA*. Estos datos son:
  - NombPlantilla: es el nombre de la plantilla.
  - TipoDoc: es el tipo de documento que se generará según la clasificación de documentos establecida (albarán, factura, etiqueta, informe).
  - Subtipo: es un número que indica cuántas veces se duplica una misma plantilla, cuando ésta se modifica y se guarda como nueva.
  - DescPlantilla: contiene una breve descripción de la plantilla para facilitar su identificación.
  - Dispositivo: es el nombre del dispositivo asociado a la plantilla.
  - NombUsuario: es el nombre del usuario que crea la plantilla.
  - FechaPlantilla: es la fecha de creación de la plantilla, que se le asigna al ser almacenada en la base de datos.
  - TamPlantilla: es el tipo de tamaño de papel correspondiente a una de los soportados por el dispositivo que tiene asignado.
  - AnchoPlantilla: es la anchura del papel, medida en milímetros.
  - AltoPlantilla: es la altura del papel, medida en milímetros.
  - TipoPapel: es el tipo de papel correspondiente a uno de los soportados por el dispositivo que tiene asignado.
  - Orientacion: indica la orientación en la que el documento será impreso, de forma vertical u horizontal.
  - Resolucion: es la resolución de impresión, correspondiente a una de las permitidas por el dispositivo.
  - Simbología: es el conjunto de caracteres disponibles, correspondiente a uno de los que soporta el dispositivo asignado.
  - Imprimible: indica si la plantilla está configurada para ser impresa o visualizada. La opción de utilizar dispositivos que no sean de impresión para que las plantillas sean visualizadas no se ha implementado, por tanto este dato no se utiliza en la aplicación actual. Se ha mantenido porque uno de los objetivos futuros es implmentar esta opción.
  - Margenes: contiene los márgenes que delimitan el área de la página en la que se pueden añadir elementos.
  - Ruta: es la ruta absoluta del directorio donde se va a almacenar el fichero de salida que genera la aplicación.
  - NumPags: número de páginas que contiene la plantilla.
  - NumElems: número total de elementos que contiene la plantilla.

- TAD *info_doc*: almacena la información de un documento cuando se solicita su impresión. Parte de su estructura es similar a la del TAD *info_plantilla*, pero su

diseño tiene como objetivo proporcionar la información necesaria para generar el fichero XML que usa el *motor de transformación*. Contiene los siguientes datos:

- OrdenPeticion: número de petición asignado por la aplicación.
- NombUsuario: nombre del usuario que solicita la impresión del documento.
- NombDoc: nombre del documento que se va a generar.
- NombPlantilla: nombre de la plantilla utilizada para impresión del documento.
- TipoDoc: es el tipo de documento que se generará según la clasificación de documentos establecida (albarán, factura, etiqueta, informe).
- Subtipo: es un número que indica cuántas veces se duplica una misma plantilla, cuando ésta se modifica y se guarda como nueva.
- Dispositivo: es el nombre del dispositivo asociado a la plantilla.
- TamPlantilla: es el tipo de tamaño de papel correspondiente a una de los soportados por el dispositivo que tiene asignado.
- AnchoPlantilla: es la anchura del papel, medida en milímetros.
- AltoPlantilla: es la altura del papel, medida en milímetros.
- TipoPapel: es el tipo de papel correspondiente a uno de los soportados por el dispositivo que tiene asignado.
- Orientacion: indica la orientación en la que el documento será impreso, de forma vertical u horizontal.
- Resolucion: es la resolución de impresión, correspondiente a una de las permitidas por el dispositivo.
- Simbologia: es el conjunto de caracteres disponibles, correspondiente a uno de los que soporta el dispositivo asignado.
- Imprimible: indica si la plantilla está configurada para ser impresa o visualizada. La opción de utilizar dispositivos que no sean de impresión para que las plantillas sean visualizadas no se ha implementado, por tanto este dato no se utiliza en la aplicación actual. Se ha mantenido porque uno de los objetivos futuros es implmentar esta opción.
- Margenes: contiene los márgenes que delimitan el área de la página en la que se pueden añadir elementos.
- Ruta: es la ruta absoluta del directorio donde se va a almacenar el fichero de salida que genera la aplicación.
- Ttel: tipo de telegrama asociado a la plantilla. Es necesario cuando las peticiones de impresión provienen de un telegrama.
- NumPags: número de páginas que contiene la plantilla.
- NumElems: número total de elementos que contiene la plantilla.

- TAD *status*: contiene la información acerca del estado de la petición actual. Sus datos son:
  - Pet: número de petición asignado por la aplicación.
  - Tipo: es el tipo de petición (atendida, desatendida).
  - Estado:

- Op: es una cadena que identifica el tipo de operación que debe realizar la aplicación.
- Subop: es una cadena que identifica, junto con el tipo de operación *Op,* las acciones a realizar por la aplicación.
- Usuario: nombre del usuario que solicita la petición.
- Plantilla: nombre de la plantilla, si se está utilizando alguna plantilla para gestionar la petición.
- Conexión: número que identifica la conexión establecida en peticiones que provieen de la *interfaz*.
- Doc: nombre del documento, si se está gestionando una petición de impresión.

A continuación se describen las estructuras que complementan al TAD *info_plantilla* y que se usan para el flujo de información que se intercambia con el *gestor de entradas*.

- TAD *info_elem*: contiene los datos comunes que comparten todos los elementos de la plantilla. Estos datos son:
  - Code: es un número único que sirve para identificar al elemento dentro de la plantilla.
  - Npag: número de página donde se encuentra el elemento.
  - TipoPos: tipo de posición del elemento: absoluta o relativa a otro elemento.
  - PosiX: posición inicial del elemento en el eje horizontal X, en milímetros.
  - PosiY: posición inicial del elemento en el eje vertical Y, en milímetros.
  - LongX: longitud del elemento en el eje horizontal X , en milímetros.
  - LongY: longitud del elemento en el eje vertical Y , en milímetros.
  - TipoE: es el tipo de elemento correspondiente a uno de los que soporta la aplicación (texto, imagen, línea, caja o código de barras).
  - NumDatos: número de datos que contiene el elemento. En los elementos de tipo texto puede tener un valor superior a 1.
  - NumLineas: número de líneas que ocupa el elemento dentro de la página.
  - PrimCol: es el número de columna donde comienza el elemento.
  - Situacion: es la disposición del elemento dentro de la página, que se divide en cabecera, cuerpo y pie.
  - Nombre: nombre del elemento asignado al crearlo.
  - Codpl: código numérico único asignado a la plantilla para su identificación.

- TAD *info_propelem*: contiene las propiedades de fuente y estilo asignadas a un elemento de tipo texto. Sus datos son:
  - Code: es un número único que sirve para identificar al elemento dentro de la plantilla.
  - Tipof: es el tipo de fuente, que hace referencia a la familia.
  - Simbolf: conjunto de caracteres soportados para el elemento de texto.
  - Tamf: es el tamaño de la fuente del elemento de texto. Se mide en puntos (*pt*).

- Altf: es la altura del elemento de texto medida en milímetros.
- Tamc: indica el número de caracteres que aparecen en una pulgada del documento. Cada pulgada son 25,4 milímetros.
- Espf: es el tipo de espaciado del elemento de texto: fijo o variable.
- EspL: indica el espaciado configurado entre las líneas del elemento de texto.
- Estf: es el estilo de la fuente (normal o cursiva).
- Densf: es la densidad de los caracteres del elemento de texto (normal o negrita).
- Subf: indica si el elemento de texto tiene la propiedad de subrayado o no.
- Colorf: color del elemento de texto.

- TAD *info_dato*: contiene la información de un dato de texto, que forma parte de una de las líneas de un elemento de texto.
  - Codd: número que identifica al dato dentro de una línea de texto por su orden de aparición.
  - Code: número único que identifica al elemento de texto dentro de la plantilla.
  - numLinea: número de línea al que pertenece el dato dentro del elemento de texto.
  - LongCar: número de caracteres que contiene el dato.
  - PosiX: posición inicial del dato en el eje horizontal X, en milímetros.
  - PosiY: posición inicial del dato en el eje vertical Y, en milímetros.
  - LongX: longitud del dato en el eje horizontal X , en milímetros.
  - NumCol: número de columna donde comienza el dato.
  - TipoD: tipo de dato (fijo o variable).
  - TipoDVar: si el dato es de tipo variable, indica el tipo de información que contiene (cadena, entero, real, fecha, hora).
  - Texto: es el contenido de un dato de tipo fijo.
  - NombDVar: es el nombre que se le asigna a un dato de tipo variable para reconocerlo dentro de la plantilla.
  - Dec: si el tipo de dato es variable y se le va a asignar información de tipo real, indica el número de decimales que puede contener el dato.

- TAD *info_imagen*: contiene la información que hace referencia a un elemento de tipo imagen. Sus datos son:
  - BytesX: es el número de píxels que contiene la imagen en el eje horizontal.
  - BytesY: es el número de píxels que contiene la imagen en el eje vertical.
  - Rotación: indica el número de grados que la imagen debe ser rotada.
  - Ruta: es la ruta absoluta del directorio donde se encuentra la imagen.

- TAD *info_borde*: contiene la información que hace referencia a los tipos de elemento línea y caja. Sus datos son:
  - Tpborde: indica si el elemento es de tipo línea o caja.
  - Grosor: indica el grosor de la línea o líneas del elemento, en milímetros.

- LongX: es la longitud del elemento en el eje horizontal X. Si el elemento es una línea vertical su valor es cero.
- LongY: es la longitud del elemento en el eje vertical Y. Si el elemento es una línea horizontal su valor es cero.
- Color: es el color de la línea o líneas del elemento.
- Fondo: si se trata de un elemento de tipo caja indica el color de fondo de la caja (blanco o negro).

- TAD *info_codb*: contiene la información relativa a un elemento de tipo código de barras. Sus datos son:
  - Dimensión: indica si el código de barras es unidimensional o de dos dimensiones.
  - Rotación: indica el número de grados que el código de barras debe ser girado.
  - Tipocb: indica el tipo de código de barras.
  - Mensaje: indica el contenido de los datos a representar en el código de barras. No todos los códigos contienen datos.
  - Fuente: si el código de barras contiene un dato de tipo *mensaje*, indica el tipo de fuente asignada al texto.
  - Altura: indica la altura del código de barras.
  - Anchura: indica la anchura de las barras del código.
  - Check: indica el tipo de dígito de *check* que presenta el código de barras. Estos dígitos sirven para comprobar la corrección de los datos de los códigos de barras.
  - Error: indica el grado de corrección que se puede aplicar al código de barras.
  - Lip: indica si el código de barras va a imprimirse con una línea de interpretación de datos o no.
  - Lipac: indica si el código de barras va a imprimirse con una línea de interpretación superior de datos o no.
  - Lipbc: indica si el código de barras va a imprimirse con una línea de interpretación inferior de datos o no.
  - Lipdc: indica si el código de barras va a imprimirse con una línea de interpretación del dígito de *check* o no.
  - Modo: indica el modo de impresión del código de barras.
  - Trunc: indica si se deben acortar las barras del centro del código de barras o eliminar parte del código.
  - Startc: representa al carácter de inicio del código de barras.
  - Stopc: representa al carácter de fin del código de barras.
  - Factor: es el factor de magnificación que se puede aplicar al código de barras.
  - Iratio: es el ratio de impresión del código de barras.
  - Wratio: es el ratio de anchura definido para las barras del código.
  - Hratio: es el ratio de altura definido para las barras del código.
  - Rratio: es el ratio definido para las filas del código de barras.

- Cratio: es el ratio definido para las columnas del código de barras.
- Nsimbol: número de símbolos que se deben añadir al final del dato de tipo *mensaje* del código de barras.
- Ntsimbol: número máximo de símbolos que puede contener el dato de tipo *mensaje* del código de barras.
- Tpdot: indica el tipo de punto o *dot* (IGP o de impresora) con el que se va a realizar la impresión del código de barras.
- Dimx: anchura de los elementos más estrechos del código en base al tamaño del tipo de punto seleccionado (*Tpdot*).
- Dimy: altura de los elementos más estrechos del código en base al tamaño del tipo de punto seleccionado (*Tpdot*).
- Formatx: número que define el formato x de código de barras de tipo *Aztec*.
- Formaty: número que define el formato y de código de barras de tipo *Aztec*.
- Appx: parámetro opcional que, junto con el valor de *Formatx,* especifica en el número de símbolos que se entrelazan unos con otros
- Appy: parámetro opcional, que junto con el valor de *Formaty*, especifica el número de símbolos que se pueden añadir al final del campo de datos.
- Dark: grado de intensidad de las barras del código.
- Hibc: es una opción que incluyen los códigos de barras de la industria para ser identificados como tales.
- Ibk: indica un formato o codificación determinada según el tipo de código de barras. No es una opción disponible para todos los tipos de códigos.
- Zip: indica si se aplica la opción de *zipper* al código de barras.
- Concat: indica una opción especial para códigos de tipo QR, junto con los tres datos siguientes. Su valor indica si hay concatenación o no de los distintos símbolos que se emplean para su codificación. En caso de haber concatenación, los tres siguientes datos indican su formato.
- Lpart: es un número que indica la longitud de las partes a concatenar.
- Npart: es un número que indica las partes totales a concatenar.
- Bpart: indica el tipo de codificación de los bytes.

Los TAD's correspondientes a las estructuras del segundo grupo, que son las diseñadas para la comunicación con el *motor de impresión,* son:

- TAD *elem_xml*: contiene los datos comunes que comparten todos los elementos de la plantilla. Estos datos son:
  - Code: es un número único que sirve para identificar al elemento dentro de la plantilla.
  - Npag: número de página donde se encuentra el elemento.
  - PosiX: posición inicial del elemento en el eje horizontal X, en milímetros.
  - PosiY: posición inicial del elemento en el eje vertical Y, en milímetros.
  - LongX: longitud del elemento en el eje horizontal X , en milímetros.
  - LongY: longitud del elemento en el eje vertical Y , en milímetros.

- TipoE: es el tipo de elemento correspondiente a uno de los que soporta la aplicación (texto, imagen, línea, caja o código de barras).
- NumLineas: número de líneas que ocupa el elemento dentro de la página.
- Nombre: nombre del elemento asignado al crearlo.

- TAD *texto_xml*: contiene la información de un dato de texto y sus propiedades. No hace referencia al elemento de texto al que pertenece al ser creado, ya que en la generación del documento a imprimir sólo se tiene en cuenta cada dato de texto por separado.
  - Cod: número que identifica al dato de tipo texto dentro de una página de la plantilla.
  - LongCar: número de caracteres que contiene el dato.
  - PosiX: posición inicial del dato en el eje horizontal X, en milímetros.
  - PosiY: posición inicial del dato en el eje vertical Y, en milímetros.
  - LongX: longitud del dato en el eje horizontal X , en milímetros.
  - NombDVar: es el nombre que se le asigna a un dato de tipo variable para reconocerlo dentro de la plantilla.
  - Contenido: es el contenido del dato de tipo texto. Si se trata de un dato de tipo variable puede que no tenga contenido si no ha sido definido en los datos de entrada del documento.
  - Tipof: es el tipo de fuente, que hace referencia a la familia.
  - Simbolf: conjunto de caracteres soportados para el elemento de texto.
  - Tamf: es el tamaño de la fuente del elemento de texto. Se mide en puntos (*pt*).
  - Altf: es la altura del elemento de texto medida en milímetros.
  - Tamc: indica el número de caracteres que aparecen en una pulgada del documento. Cada pulgada son 25,4 milímetros.
  - Espf: es el tipo de espaciado del elemento de texto: fijo o variable.
  - EspL: indica el espaciado configurado entre las líneas del elemento de texto.
  - Estf: es el estilo de la fuente (normal o cursiva).
  - Densf: es la densidad de los caracteres del elemento de texto (normal o negrita).
  - Subf: indica si el elemento de texto tiene la propiedad de subrayado o no.
  - Colorf: color del elemento de texto.

- TAD *imagen_xml*: contiene la información que hace referencia a un elemento de tipo imagen. Sus datos son:
  - BytesX: es el número de píxels que contiene la imagen en el eje horizontal.
  - BytesY: es el número de píxels que contiene la imagen en el eje vertical.
  - Rotación: indica el número de grados que la imagen debe ser rotada.
  - Ruta: es la ruta absoluta del directorio donde se encuentra la imagen.

- TAD borde_xml: contiene la información que hace referencia a los tipos de elemento línea y caja. Sus datos son:

- Tpborde: indica si el elemento es de tipo línea o caja.
- Grosor: indica el grosor de la línea o líneas del elemento, en milímetros.
- LongX: es la longitud del elemento en el eje horizontal X. Si el elemento es una línea vertical su valor es cero.
- LongY: es la longitud del elemento en el eje vertical Y. Si el elemento es una línea horizontal su valor es cero.
- Color: es el color de la línea o líneas del elemento.
- Fondo: si se trata de un elemento de tipo caja indica el color de fondo de la caja (blanco o negro).

- TAD codb_xml: contiene la información relativa a un elemento de tipo código de barras. Sus datos son:
  - Dimensión: indica si el código de barras es unidimensional o de dos dimensiones.
  - Rotación: indica el número de grados que el código de barras debe ser girado.
  - Tipocb: indica el tipo de código de barras.
  - Mensaje: indica el contenido de los datos a representar en el código de barras. No todos los códigos contienen datos.
  - Fuente: si el código de barras contiene un dato de tipo *mensaje,* indica el tipo de fuente asignada al texto.
  - Altura: indica la altura del código de barras.
  - Anchura: indica la anchura de las barras del código.
  - Check: indica el tipo de dígito de *check* que presenta el código de barras. Estos dígitos sirven para comprobar la corrección de los datos de los códigos de barras.
  - Error: indica el grado de corrección que se puede aplicar al código de barras.
  - Lip: indica si el código de barras va a imprimirse con una línea de interpretación de datos o no.
  - Lipac: indica si el código de barras va a imprimirse con una línea de interpretación superior de datos o no.
  - Lipbc: indica si el código de barras va a imprimirse con una línea de interpretación inferior de datos o no.
  - Lipdc: indica si el código de barras va a imprimirse con una línea de interpretación del dígito de *check* o no.
  - Modo: indica el modo de impresión del código de barras.
  - Trunc: indica si se deben acortar las barras del centro del código de barras.
  - Startc: representa al carácter de inicio del código de barras.
  - Stopc: representa al carácter de fin del código de barras.
  - Factor: es el factor de magnificación que se puede aplicar al código de barras.
  - Iratio: es el ratio de impresión del código de barras.
  - Wratio: es el ratio de anchura definido para las barras del código.
  - Hratio: es el ratio de altura definido para las barras del código.

- Rratio: es el ratio definido para las filas del código de barras.
- Cratio: es el ratio definido para las columnas del código de barras.
- Nsimbol: número de símbolos que se deben añadir al final del dato de tipo *mensaje* del código de barras.
- Ntsimbol: número máximo de símbolos que puede contener el dato de tipo *mensaje* del código de barras.
- Dimx: anchura de los elementos más estrechos del código en base al tamaño del tipo de punto seleccionado (*Tpdot*).
- Dimy: altura de los elementos más estrechos del código en base al tamaño del tipo de punto seleccionado (*Tpdot*).
- Formatx: número que define el formato x de código de barras de tipo *Aztec*.
- Formaty: número que define el formato y de código de barras de tipo *Aztec*.
- Appx: parámetro opcional que, junto con el valor de *Formatx*, especifica en el número de símbolos que se entrelazan unos con otros
- Appy: parámetro opcional, que junto con el valor de *Formaty*, especifica el número de símbolos que se pueden añadir al final del campo de datos.
- Tpdot: indica el tipo de punto o *dot* (IGP o de impresora) con el que se va a realizar la impresión del código de barras.
- Dark: grado de intensidad de las barras del código.
- Hibc: es una opción que incluyen los códigos de barras de la industria para ser identificados como tales.
- Ibk: indica un formato o codificación determinada según el tipo de código de barras. No es una opción disponible para todos los tipos de códigos.
- Zip: indica si se aplica la opción de *zipper* al código de barras.
- Concat: indica una opción especial para códigos de tipo QR, junto con los tres datos siguientes. Su valor indica si hay concatenación o no de los distintos símbolos que se emplean para su codificación. En caso de haber concatenación, los tres siguientes datos indican su formato.
- Lpart: es un número que indica la longitud de las partes a concatenar.
- Npart: es un número que indica las partes totales a concatenar.
- Bpart: indica el tipo de codificación de los bytes.

El resto de estructuras que se emplean son listas que se usan cuando se desea obtener la lista de plantillas del usuario o la lista de dispositivos disponibles junto con sus listas de opciones de impresión. Los datos que guardan estas listas tiene siempre formato de tipo cadena en C.

C.1.2 Módulo gestor de entradas

Previamente se había definido como el módulo que gestiona las peticiones de entrada que llegan al sistema. Las estructuras que maneja han sido detalladas en la sección anterior y ahora se van a explicar algunos de sus detalles internos.

Se compone de un servidor concurrente que atiende simultáneamente las peticiones que recibe a través de procesos esclavos o hijos. La información acerca de las peticiones y su estado la solicita al gestor de la base de datos. De este modo conoce el estado de todas ellas y en caso de ocurrir algún evento inesperado puede recuperar su información. La importancia de conocer el estado de las peticiones se debe a que funciona como una máquina de estados. Esto permite identificar estados erróneos y la posibilidad de acceder sólo a los estados permitidos para evitar flujos de ejecución no deseados. El estado viene determinado por el tipo de operación a realizar. Se debe comprobar el estado anterior para ver que la operación es factible. La operación a realizar se indica en los mensajes de datos que provienen desde la *interfaz* mediante una o dos palabras claves que componen la cabecera del mensaje junto con el número de petición. Estos mensajes son cifrados siguiendo un patrón de ordenación de datos y utilizando separadores predefinidos. La cabecera del mensaje es la que permite descifrar su contenido de forma correcta, ya que es la que determina su formato. El tipo de operación le indica si el mensaje contiene información que debe almacenar o si debe enviar información a la *interfaz*. Una vez que termina la operación que se le ha solicitado debe comunicar si ha sido realizada con éxito. Para ello se envía una palabra clave que hace referencia al tipo de operación realizada y si se ha completado correctamente. Este mensaje de confirmación es similar al que se envían clientes y servidores para comprobar si la conexión ha tenido éxito. Cuando se recibe una petición de impresión se obtiene toda la información necesaria en una única conexión y es el *motor de impresión* el que se encarga de su gestión. Las operaciones de este tipo de peticiones se ejecutan secuencialmente siguiendo un orden predefinido por el módulo que las gestiona.

C.1.3 Módulo motor de impresión

Los componentes de este módulo realizan una serie de operaciones cuyo fin es la obtención del fichero de salida traducido al lenguaje de impresión del dispositivo destino. El primer componente que interviene en una petición de impresión es el *gestor XML*, obteniendo los datos de las estructuras definidas en la sección del módulo *gestor de base de datos*. Su función es generar el fichero XML con la información del documento a imprimir. Para ello funciona como un intérprete de XML, encargándose de insertar, eliminar, modificar o consultar los elementos que forman el fichero. Se ayuda del *parser Expat*, que ha sido implementado para utilizarlo en la plataforma específica C. La estructura que sigue el fichero XML se explica a continuación.

Este tipo de ficheros contiene una estructura de *tags* o etiquetas que delimitan uno o más elementos. Se pueden definir distintos niveles de etiquetas y pueden ser vacías o contener más datos. Su estructura es comparable a la de los árboles, que contiene una serie de nodos ordenados por distintos niveles y que pueden almacenar cierta información. En este caso los nodos serían las etiquetas.

El fichero comienza con una serie de *tags* que indican el nombre del documento, de la plantilla, del usuario, del dispositivo y ciertas características importantes acerca de la plantilla: tipo de papel, tamaño, orientación, márgenes, conjunto de símbolos permitidos. Toda esta información se obtiene de las estructuras *info_plantilla* e *info_doc*.

A continuación se encuentran las opciones de impresión seleccionadas. Cada opción tiene un *tag* con el nombre que la identifica y otro con el valor elegido. Esta información la obtiene de las estructuras en forma de listas que se usan para las opciones de impresión de los dispositivos. Estas listas son obtenidas por el *gestor XML*, consultando el fichero XML de capacidades del dispositivo.

Finalmente se encuentra un *tag*, que engloba toda la estructura de páginas del documento, indicando el número total a imprimir y añadiendo un nuevo nivel de *tags* por cada página, con un atributo numérico que las identifica. Cada una de estas páginas añade un nuevo nivel para indicar los elementos que contiene. Cada elemento se identifica con un número y especifica el tipo de dato como un atributo dentro del *tag*. Estos datos, a su vez, añaden nuevos niveles de *tags* que hacen referencia a la posición absoluta y el área que ocupan en la página, así como al contenido, dependiendo del tipo de dato. Tanto las páginas como los elementos que contienen se sitúan en el fichero por orden de aparición en el documento. En el caso de un elemento de tipo texto se especifica la cadena y el número de caracteres. Las líneas indican su orientación, grosor y longitud. Las cajas indican también su grosor y la longitud de sus bordes. Las imágenes indican la ruta del archivo que las contiene, el número de bytes por fila y el número de columnas. Los códigos de barra indican el tipo de código y según el tipo todas sus propiedades necesarias para su impresión. Toda esta información se obtiene de las estructuras: *elem_xml*, *texto_xml*, *imagen_xml*, *borde_xml* y *codb_xml*.

Esta es la estructura general para imprimir un solo documento. En el caso de dispositivos que pueden imprimir varios documentos distintos en la misma petición, la estructura inicial es la misma, pero en el momento de especificar la estructura de páginas a imprimir se indica sólo el contenido de los elementos fijos de la plantilla. Esto significa que el contenido de los elementos de texto variables de cada documento a imprimir se indicará a continuación en un grupo de *tags*, identificando cada documento con un número. Esos *tags* se subdividirán en nuevos niveles que indicarán el número de dato variable al que se refieren y su contenido. Esto es posible asignando a todos los elementos de texto variables un número único por orden de aparición en la plantilla.

El *gestor de imágenes* se encarga de convertir las imágenes a distintos formatos y gestiona las estructuras de cabecera de los archivos *bitmap* y *jpeg*. Entre otros datos necesita obtener el número de filas y el número de píxeles que contiene por fila. Decodifica esta información y la vuelve a codificar para que pueda incluirse en el fichero XML.

El *gestor de capacidades* también funciona como un intérprete de XML, consultando la información de este fichero y de los ficheros XML de capacidades de los dispositivos.

Debe comprobar las funcionalidades del dispositivo de impresión y examinar todos los elementos que forman el documento para ver si es posible su correcta impresión. La estructura de los ficheros de capacidades del dispositivo comienza con una serie de *tags* que describen sus características generales, obtenidas de hoja de especificación que proporcionan los fabricantes. El resto del fichero se compone de todas las opciones de impresión identificadas por un número. Cada opción de impresión contiene su nombre, su valor por defecto, su valor de fábrica y una lista de *tags* con todos sus valores disponibles.

Para ver cómo es la estructura de un esquema XSD se detalla a continuación el correspondiente a los ficheros XML de capacidades.

```
<?xml version="1.0" encoding="UTF-8"?>
<xsd:schema xmlns:xsd="http://www.w3.org/2001/XMLSchema" elementFormDefault="qualified">
<xsd:include schemaLocation="tipos.xsd" />
  <xsd:element name="impresora">
    <xsd:complexType>
      <xsd:attribute name="id" type="xsd:string" />
        <xsd:all>
          <xsd:element name="fabricante" type="xsd:string" />
           <xsd:element name="modelo" type="xsd:string" />

           <!-- mecanismo -->

           <xsd:element name="mecanismo">
              <xsd:annotation>
                <xsd:documentation>
                        Esta seccion describe las distintas facetas de la impresora
                        segun su mecanismo: como el tipo de impresora (tipos soportados:
                         laser, led, impacto, sublimación, transferencia) y la máxima
                        resolucion.
                 </xsd:documentation>
               </xsd:annotation>
               <xsd:complexType>
                  <xsd:all>
                     <xsd:element name="laser" minOccurs="0">
                        <xsd:complexType />
                      </xsd:element>
                     <xsd:element name="led" minOccurs="0">
                        <xsd:complexType />
                     </xsd:element>
                     <xsd:element name="tinta" minOccurs="0">
                        <xsd:complexType />
                     </xsd:element>
                     <xsd:element name="matricial" minOccurs="0">
                        <xsd:complexType />
                     </xsd:element>
                     <xsd:element name="impacto" minOccurs="0">
                        <xsd:complexType />
                     </xsd:element>
                     <xsd:element name="sublimacion" minOccurs="0">
                        <xsd:complexType />
                     </xsd:element>
                     <xsd:element name="transferencia" minOccurs="0">
                        <xsd:complexType />
                     </xsd:element>
                     <xsd:element name="termal" minOccurs="0">
```

```
            <xsd:complexType />
          </xsd:element>
          <xsd:element name="resolucion" minOccurs="0">
            <xsd:complexType>
              <xsd:all>
                <xsd:element name="dpi" minOccurs="0">
                 <xsd:complexType>
                   <xsd:all>
                     <xsd:element name="x" type="xsd:integer" minOccurs="0" />
                      <xsd:element name="y" type="xsd:integer" minOccurs="0" />
                   </xsd:all>
                 </xsd:complexType>
                </xsd:element>
              </xsd:all>
            </xsd:complexType>
          </xsd:element>
        </xsd:all>
      </xsd:complexType>
    </xsd:element>

    <!-- lenguajes -->

    <xsd:element name="lenguaje" minOccurs="0">
          <xsd:annotation>
            <xsd:documentation>
                  Esta seccion describe los lenguajes de impresion que
                  entiende la impresora. Normalmente tendra mas de uno.
            </xsd:documentation>
          </xsd:annotation>
            <xsd:complexType>
              <xsd:all>
                <xsd:element name="zpl" minOccurs="0">
                    <xsd:complexType>
                        <xsd:attribute name="level" type="xsd:string" />
                    </xsd:complexType>
                  </xsd:element>
                  <xsd:element name="pcl" minOccurs="0">
                    <xsd:complexType>
                      <xsd:attribute name="level" type="xsd:string" />
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="hpgl2" minOccurs="0">
                    <xsd:complexType>
                      <xsd:attribute name="level" type="xsd:string" />
                    </xsd:complexType>
                </xsd:element>
                <xsd:element name="pjl" minOccurs="0">
                    <xsd:complexType />
                 </xsd:element>
                <xsd:element name="pgl" minOccurs="0">
                    <xsd:complexType />
                </xsd:element>
                <xsd:element name="text" minOccurs="0">
                    <xsd:complexType>
                      <xsd:all>
                          <xsd:element name="charset" type="xsd:string" >
                      </xsd:all>
                    </xsd:complexType>
                 </xsd:element>
              </xsd:all>
```

```
  </xsd:complexType>
</xsd:element>

<!-- funcionalidad -->

<xsd:element name="funcionalidad">
        <xsd:annotation>
          <xsd:documentation>
                Esta seccion describe los lenguajes de impresion que
                entiende la impresora. Normalmente tendra mas de uno.
          </xsd:documentation>
        </xsd:annotation>
        <xsd:complexType>
          <xsd:attribute name="varios" type="xsd:nonNegativeInteger" />
            <xsd:all>
              <xsd:element name="Texto" minOccurs="0">
                 <xsd:complexType />
              </xsd:element>
              <xsd:element name="Imagen" minOccurs="0">
                 <xsd:complexType />
              </xsd:element>
              <xsd:element name="Color" minOccurs="0">
                 <xsd:complexType />
              </xsd:element>
              <xsd:element name="Negro" minOccurs="0">
                 <xsd:complexType />
              </xsd:element>
              <xsd:element name="CodBarras" minOccurs="0">
                  <xsd:complexType />
              </xsd:element>
              <xsd:element name="Normal" minOccurs="0">
                  <xsd:complexType />
              </xsd:element>
              <xsd:element name="Multifuncion" minOccurs="0">
                  <xsd:complexType />
              </xsd:element>
            </xsd:all>
        </xsd:complexType>
</xsd:element>

<!-- capacidades de la impresora -->

<xsd:element name="capacidades" minOccurs="0">
   <xsd:complexType>
     <xsd:sequence>
       <xsd:element name="capacidad" type="xsd:string" />
         <xsd:complexType>
           <xsd:all>
             <xsd:element name="arg_longname" type="comentario" />
             <xsd:element name="arg_shortname" type="comentario" />
             <xsd:element name="arg_execution">
               <xsd:complexType>
                 <xsd:all>
                   <xsd:element name="arg_grupo" type="xsd:string"/>
                   <xsd:element name="arg_orden" type="xsd:int" />
                 </xsd:all>
               </xsd:complexType>
             </xsd:element>
             <xsd:element name="arg_max" type="xsd:float"  />
```

```
                                        <xsd:element name="arg_min" type="xsd:float" />
                                        <xsd:element name="arg_shortname_false"
                                                     type="comentario/>
                                        <xsd:element name="arg_fabrica" type="xsd:int" />
                                        <xsd:element name="arg_defval" type="xsd:string" />
                                        <xsd:element name="enum_vals" minOccurs="0">
                                          <xsd:complexType>
                                            <xsd:sequence>
                                              <xsd:element name="enum_val"
                                                           maxOccurs="unbounded">
                                                <xsd:complexType>
                                                  <xsd:all>
                                                    <xsd:element name="ev_longname"
                                                                 type="comentario" />
                                                    <xsd:element name="ev_shortname"
                                                                 type="comentario" />
                                                    <xsd:element name="ev_driverval">
                                                      <xsd:complexType>
                                                        <xsd:all>
                                                          <xsd:element name="arg_pcl"
                                                                type="xsd:string" minOccurs="0"/>
                                                          <xsd:element name="arg_pjl"
                                                                type="xsd:string" minOccurs="0"/>
                                                          <xsd:element name="arg_hpgl2"
                                                                 type="xsd:string"minOccurs="0"/>
                                                          <xsd:element name="arg_zpl"
                                                                 type="xsd:string"minOccurs="0"/>
                                                          <xsd:element name="arg_pgl"
                                                                  type="xsd:string"  minOccurs="0"/>
                                                        </xsd:all>
                                                      </xsd:complexType>
                                                    </xsd:element>
                                                  </xsd:all>
                                                  <xsd:attribute name="id" type="evalID"
                                                                 use="required"/>
                                                </xsd:complexType>
                                              </xsd:element>
                                            </xsd:sequence>
                                          </xsd:complexType>
                                        </xsd:element>
                                      </xsd:all>
                                      <xsd:attribute name="tipo" type="capacidadTipos" use="required" />
                                    <xsd:attribute name="id" type="capacidadID" use="required" />
                                  </xsd:complexType>
                                </xsd:sequence>
                              </complexType>
                            </xsd:element>

                            <!-- fin de las capacidades de la impresora -->

                            <xsd:element name="contrib_url" type="xsd:string" minOccurs="0" />
                            <xsd:element name="comentario" type="i18nString" minOccurs="0" />
                          </xsd:all>
                        <xsd:attribute name="id" type="printerID" use="required" />
                      </xsd:complexType>
                    </xsd:element>
                  </xsd:schema>
```

Una vez que el *gestor de capacidades* termina, el *motor de transformación* realiza la traducción final del documento, invocando al *parser* adecuado. Como cualquier intérprete, este *parser* se encarga de analizar la información del fichero XML y añade los parámetros adecuados a los comandos que usa el lenguaje nativo del dispositivo. La transformación final del fichero generado se realiza mediante XSLT, que es un lenguaje de transformación incluido en la familia de lenguajes que forman XSL. Los lenguajes de impresión soportados y sus comandos fueron descritos brevemente en el anexo A. Para una mayor información acerca de todos sus comandos se debe consultar los manuales que proporcionan sus fabricantes.

C.2 Diseño de objetos

Esta sección detalla el diseño de los objetos diseñados para el módulo *interfaz*, que forma parte de la aplicación *front-end*. El módulo *interfaz* ofrece operaciones de dos tipos:

- Operaciones de gestión de los componentes: son aquellas que se usarán para la creación, organización y destrucción de todos aquellos componentes de la interfaz como botones, tablas, listas, formularios, etc. Estas operaciones son realizadas por las clases de tipo ventana .

- Operaciones de gestión de eventos: son aquellas operaciones que se encargan de recibir los posibles eventos generados y realizar las diferentes operaciones que se hayan asignado a cada uno de ellos. Los eventos principalmente serán los que se generen cuando el usuario interactúe con los componentes de la interfaz. Estas operaciones son realizadas por los gestores de ventana.

Las figuras C.1, C.2 y C.3 muestran las clases de tipo ventana y las figuras C.4 y C.5 las que corresponden a sus gestores. Las ventanas de la figura C.1 son las que recogen y presentan la información relativa a la impresión de documentos y al usuario. Las ventanas de la figura C.2 son las que recogen la información para el inicio de creación de las plantillas y las de la figura C.3 completan esta funcionalidad mediante la creación de sus elementos. Es una representación más detallada de las clases respecto a la que se hizo en capítulo 3 en la figura 3.7. Estas figuras detallan las propiedades y operaciones que pueden realizar sus objetos. Debido a sus extensión se han dividido en más de una figura para poder visualizar mejor su contenido.

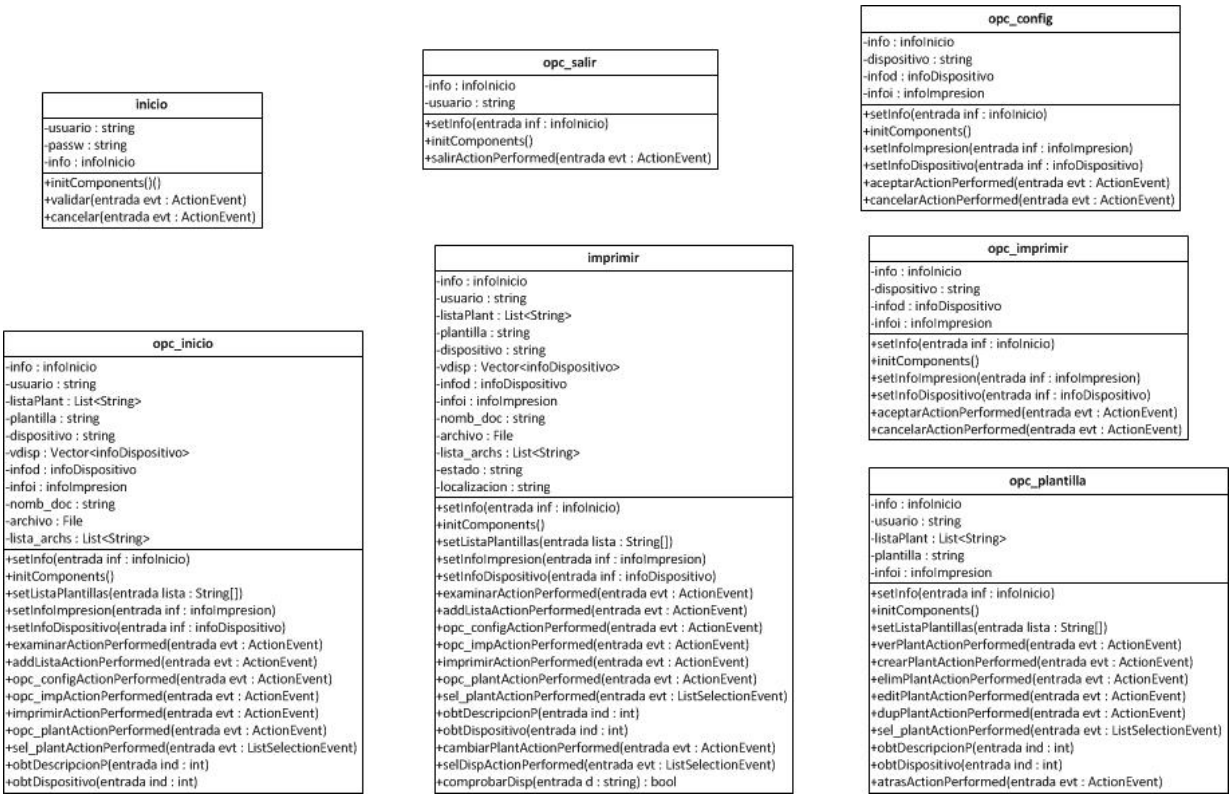

Figura C.1 Clases de las ventanas para la impresión de documentos

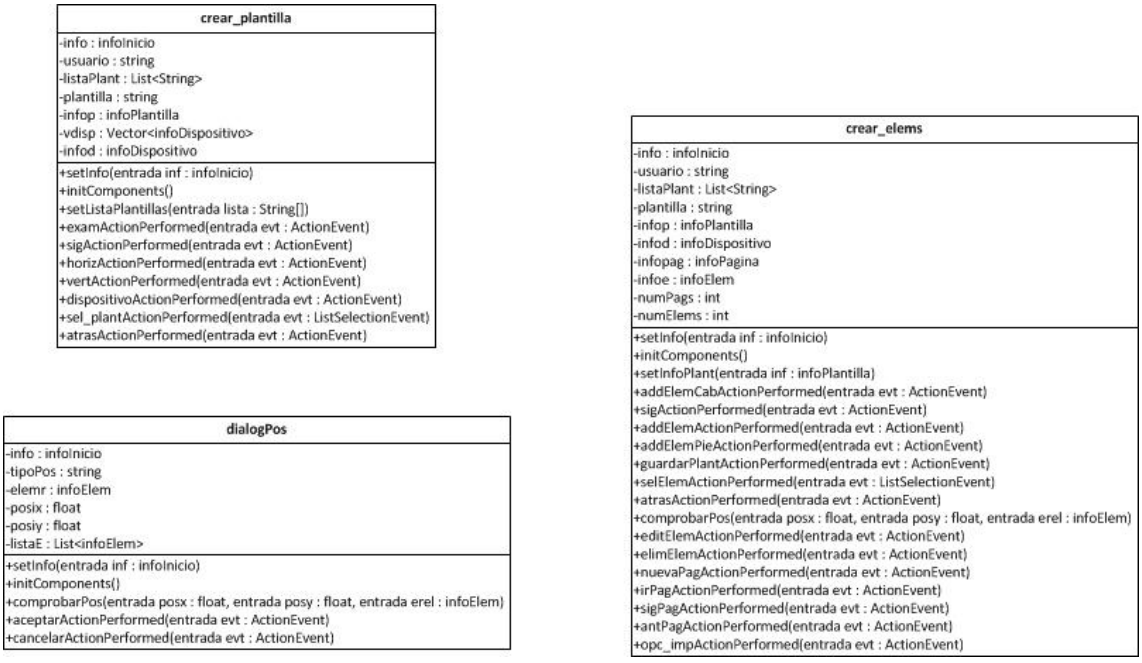

Figura C.2 Clases de las ventanas de inicio de creación de plantillas

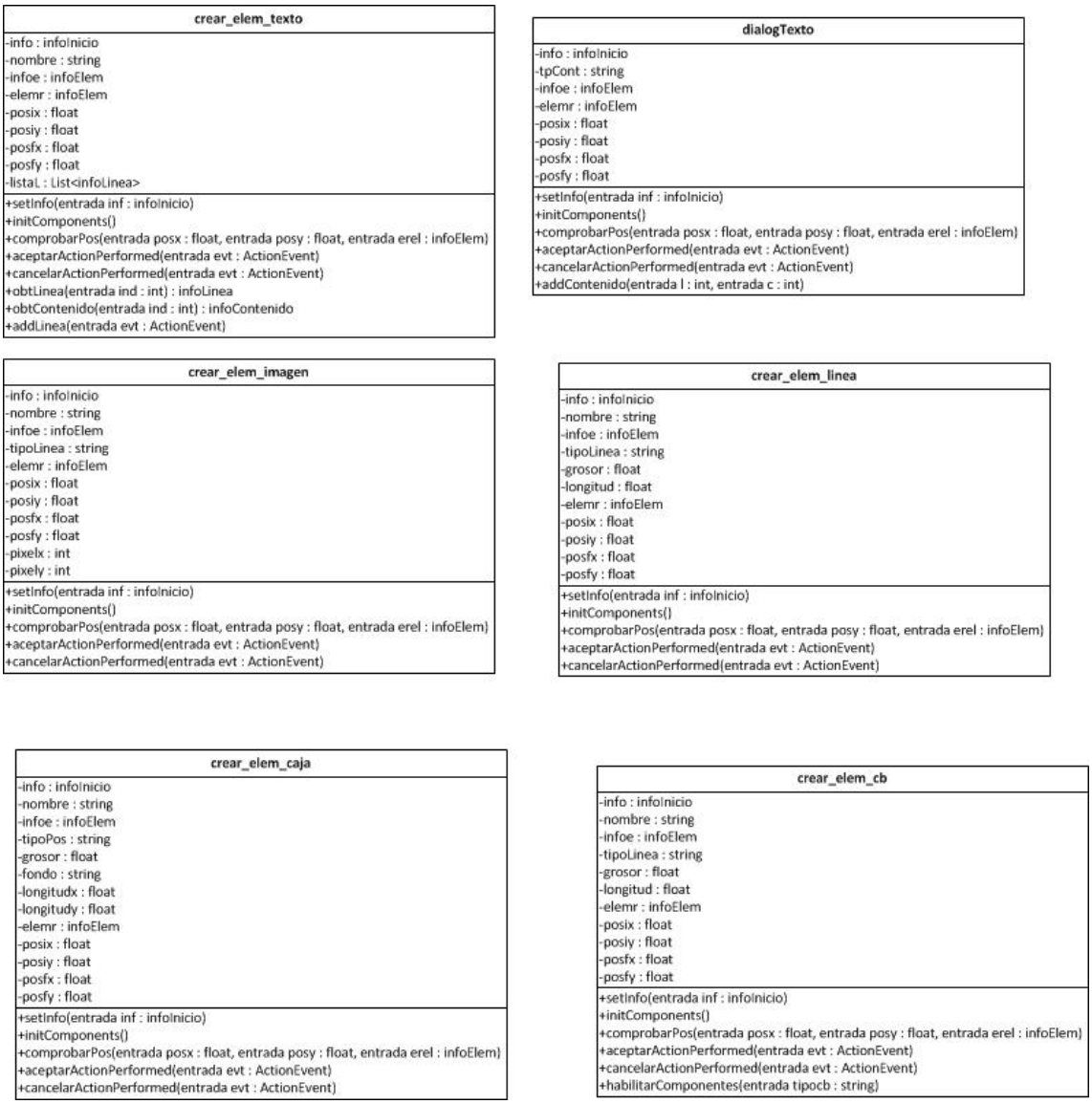

Figura C.3 Clases de las ventanas de creación de elementos de las plantillas

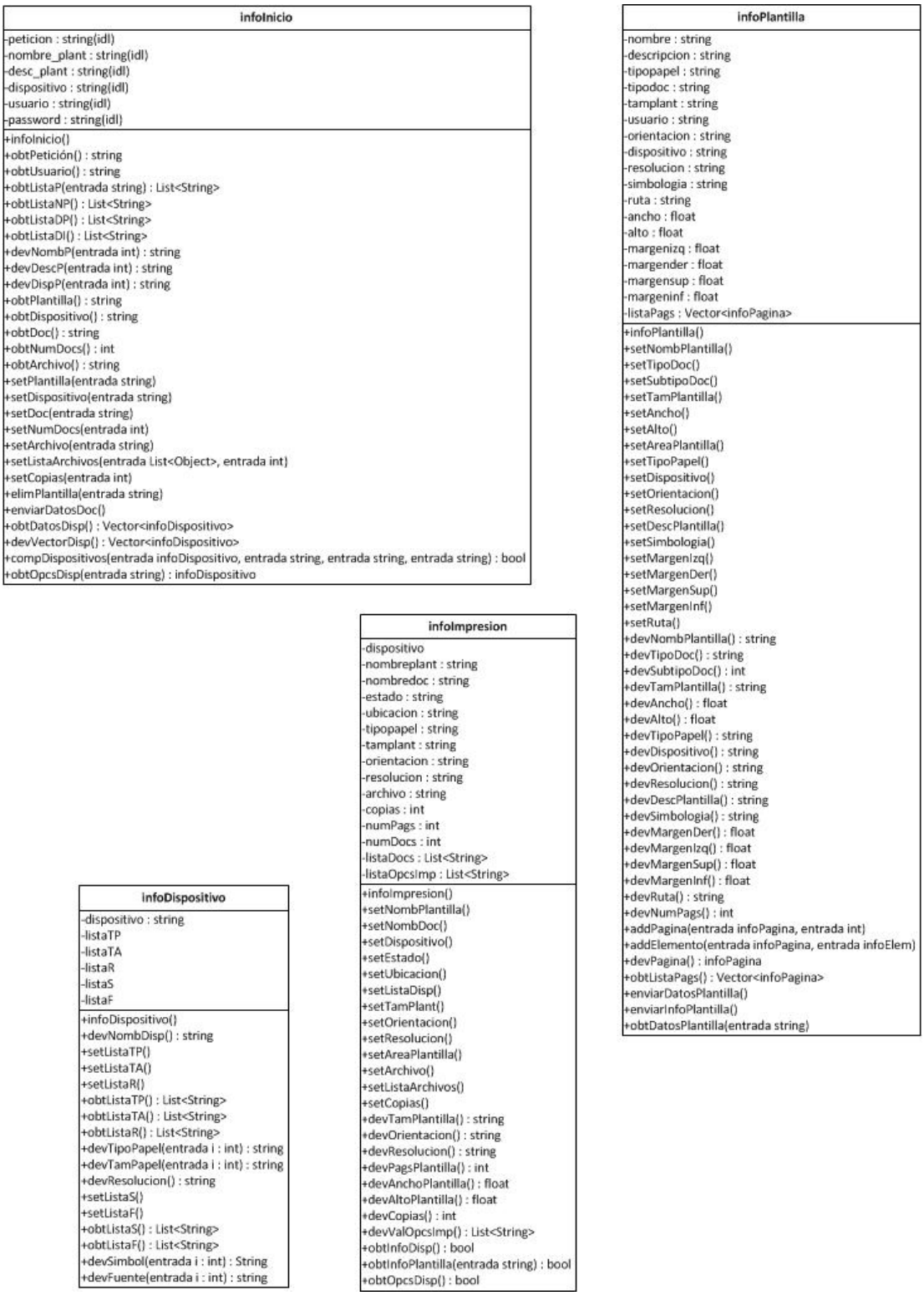

Figura C.4 Clases de los gestores de ventana

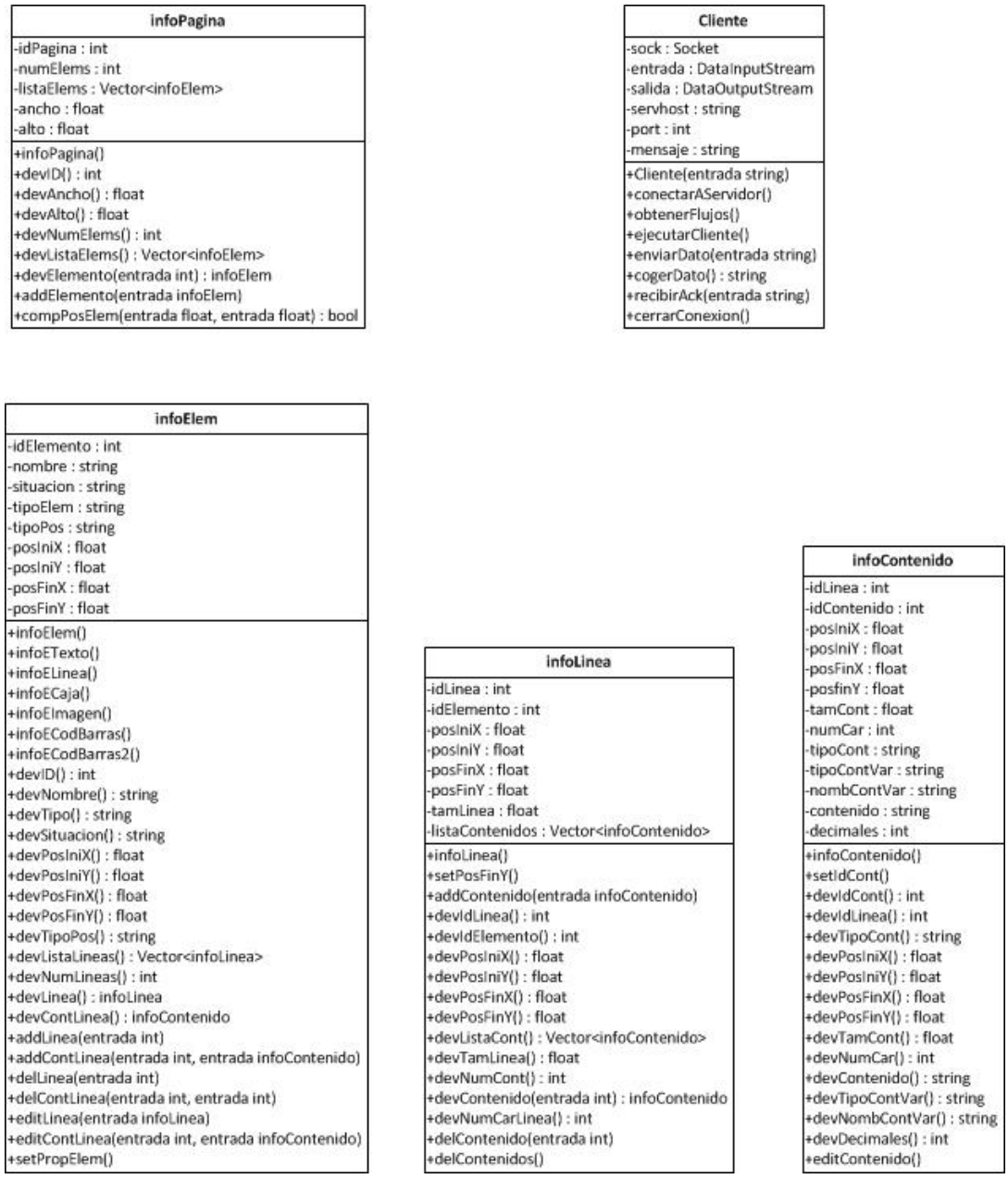

Figura C.5 Clases de los gestores de ventana restantes

Las clases de los *gestores de ventana* fueron descritas al final del capítulo 3. Para comprender mejor su comportamiento las figuras C.6, C.7 y C.8 muestran nuevas visiones dinámicas que incluyen estos objetos. En el capítulo 2 se mostraba el comportamiento del sistema sin detallar algunos componentes para evitar la complejidad que suponía entender su funcionamiento. En esta parte del documento, como ya se han presentado todos los objetos, se pueden mostrar unos esquemas algo más completos.

En la figura C.6 muestra la secuencia de eventos en la identificación de un usuario a través de la interfaz, tal y como se mostraba en la figura 3.3 pero con los objetos de la

interfaz. En la figura se ve cómo el usuario introduce los datos y la ventana de inicio crea un objeto *infoInicio* para que guarde esos datos y los gestione. En este caso el objeto *infoInicio* recibe los datos del usuario y crea uno de la clase *Cliente* para que envíe estos datos al *gestor de entradas*. El *gestor de entradas* debe aceptar la conexión y recibir los datos, identificando su cabecera para ver qué tipo de mensaje es y qué información contiene. Una vez que reconoce el tipo de operación que debe realizar, lleva a cabo la acción adecuada, siendo en este caso la comprobación del usuario. Se comunica con el *gestor de la base de datos* y le envía los datos del usuario para que los compruebe. Deberá notificarle al *gestor de entradas* si el usuario es correcto o no y en caso de ser correcto, el *gestor de base de datos* le devuelve la lista de plantillas que posee ese usuario, junto con la descripción de esas plantillas y el dispositivo al que están asociadas. Una vez obtenidos todos los datos, el *gestor de entradas* los enviará por medio del socket que mantiene abierto mientras el objeto *Cliente* espera recibir dichos datos. Al recibir los datos, el objeto *infoInicio* decodifica el mensaje recibido y si el usuario es correcto, guardará la lista de plantillas, sus descripciones y dispositivos y ordenará al *Cliente* el cierre de la conexión. En este caso se acepta al usuario y se muestra una ventana con el nombre del usuario y los datos acerca de sus plantillas. Si el usuario no fuera correcto se mostraría un mensaje de error y éste debería identificarse de nuevo.

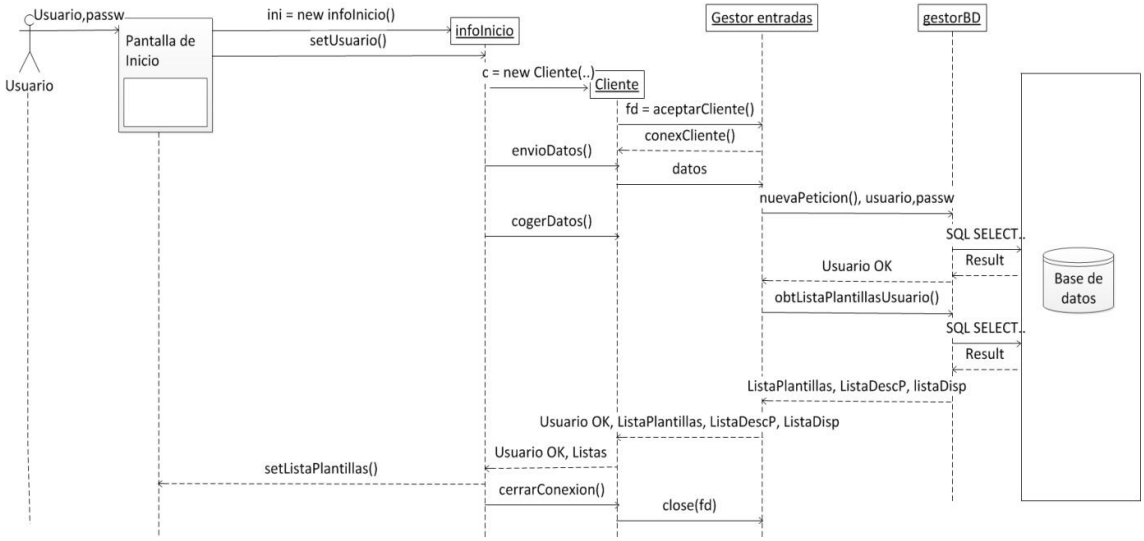

Figura C.6 Diagrama de secuencia de la identificación de usuarios

El siguiente caso de uso representado en la figura C.7 muestra cómo es la creación de una plantilla, al igual que se mostraba en el diagrama resumido de la figura 3.4 del capítulo 3. En este caso se ilustran las acciones principales que implican la creación de la plantilla. El punto de partida es la pantalla de inicio que muestra la lista de plantillas y las distintas opciones que puede realizar el usuario, una vez que se ha identificado correctamente.

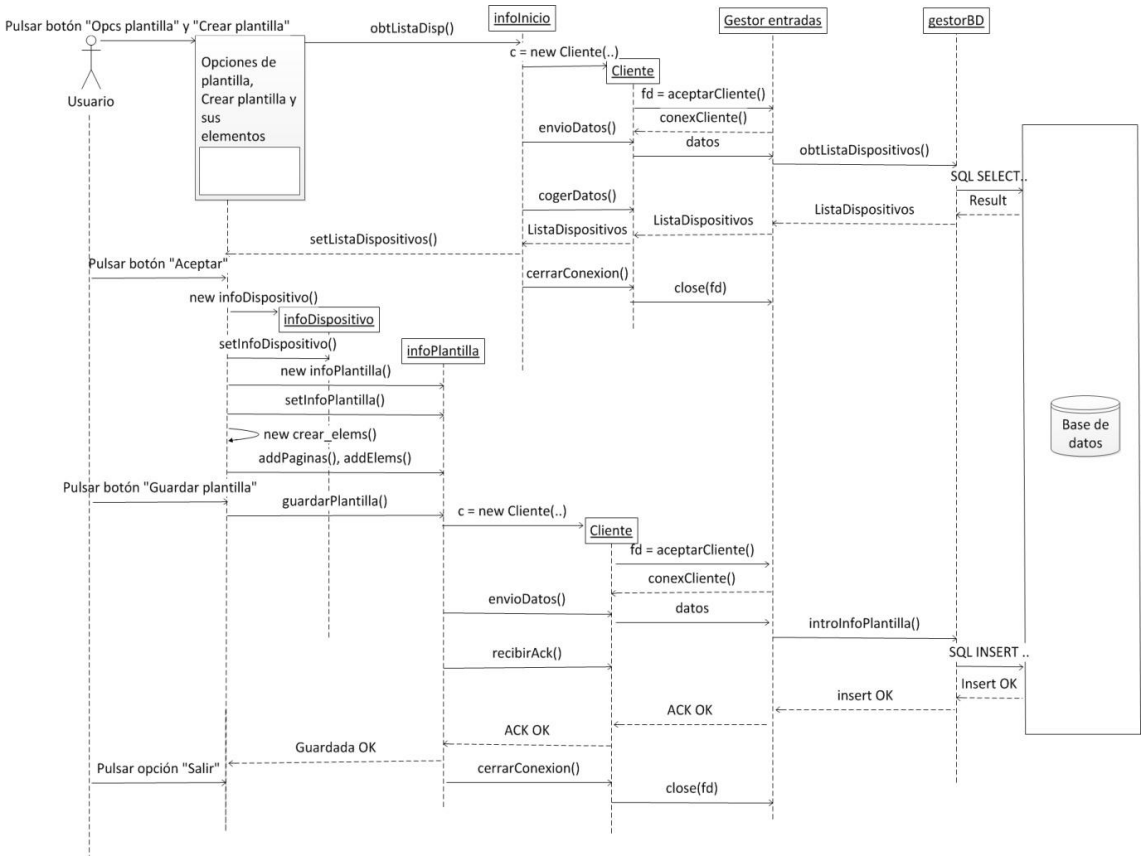

Figura C.7 Diagrama de secuencia de la creación de plantillas.

La figura C.7 agrupa el conjunto de ventanas que van apareciendo en una sola. Como se aprecia en la figura, el usuario debe pulsar una serie de botones para avanzar hasta la ventana que muestre las primeras opciones para crear la plantilla. Antes de que aparezca esta ventana se usa el objeto *infoInicio*, creado previamente, para obtener parte de la información a mostrar. Se crea de nuevo un objeto *Cliente* para que envíe un mensaje al gestor de entradas indicándole que debe obtener la lista de dispositivos disponibles en la base de datos. Además de enviar la lista de dispositivos, se incluirán también algunas de sus capacidades como sus tipos de papel, sus tamaños, sus resoluciones, sus conjuntos de símbolos y sus fuentes. Toda esta información es mandada por medio del socket y la recibe *infoInicio* a través del *Cliente*. Esta información se guarda y se muestra al usuario en la ventana *Crear plantilla*. El usuario debe rellenar todos los campos con la información necesaria como por ejemplo el nombre de la plantilla, su tamaño, tipo de documento, descripción, dispositivo,..etc. Al pulsar el botón "Aceptar" se guarda toda esta información y se muestra la siguiente ventana *Crear elems*. Se crean dos objetos nuevos para gestionar la información: *infoPlantilla* guarda la información relativa a las plantillas e *infoDispositivo* la relativa al dispositivo elegido y sus capacidades. La nueva ventana *Crear elems* permite la creación de los elementos de la plantilla, que se gestionan a través de nuevas ventanas junto con el objeto *infoPlantilla*, que es el que va

guardando toda la información. Este objeto se encarga de comprobar las posiciones y dimensiones de todos los elementos para que no interfieran unos con otros y de esta forma permite o no su creación. El usuario puede ir añadiendo nuevas páginas y elementos a la plantilla. Todos los elementos creados en la página actual de la plantilla van apareciendo en la ventana *Crear elems* en forma de lista. Una vez que el usuario termina la creación de la plantilla pulsa el botón "Guardar plantilla", tal y como muestra la figura C.7, y es *infoPlantilla* el que invoca al *Cliente* para enviar toda la información de la plantilla al *gestor de entradas*, a través del socket. El gestor de entradas reconoce el tipo de operación a realizar y organiza la información de la plantilla en estructuras de datos para enviársela al *gestor de la base de datos* y que éste pueda almacenarla en las tablas disponibles para ello. Si la inserción ha sido correcta, se le comunicará al *gestor de entradas* y éste enviará un mensaje de confirmación al *Cliente*, que al recibirlo cerrará la conexión. El usuario puede continuar con la impresión de un documento desde la ventana *Crear elems* o bien salir de la aplicación. Debido al exceso de operaciones a realizar, el caso de uso termina con el cierre de la aplicación y se continúa su ejecución en el siguiente caso de uso.

La figura C.8 es la nueva versión detallada de la figura 3.5, que aparecía en el capítulo 3. Se muestra la secuencia de eventos correspondiente al caso de uso de impresión de documentos utilizando la plantilla creada y comienza desde la ventana *Crear elems* del caso anterior. Para que el esquema de la figura C.8 no resulte demasiado extenso se han agrupado las ventanas *Crear elems* e *Imprimir* en una sola y también algunas funciones. El usuario va solicitando o enviando parámetros de impresión, al igual que se hacía en el caso anterior. Cuando el usuario pulsa el botón "Opciones de impresión" se crea la nueva ventana *Imprimir*. Al mismo tiempo se crea un objeto de la clase *infoImpresion*, que será el encargado de obtener cierta información del dispositivo y sus capacidades. Al igual que en los casos anteriores se necesita invocar un objeto *Cliente* para que envíe el mensaje al *gestor de entradas* y éste gestione las operaciones a realizar. El *gestor de entradas* solicita la información al *gestor de la base de datos*, proporcionándole el nombre del dispositivo, y éste le devuelve su estado, ubicación, lenguajes nativos de impresión y un conjunto de listas que contienen una serie valores correspondientes a sus capacidades, así como el valor seleccionado por defecto para cada una de ellas. El *gestor de entradas* codifica toda esta información y se la envía al *Cliente*, siendo decodificada y gestionada por *infoImpresion* para que pueda ser mostrada en las distintas ventanas que se muestran en la figura. En la ventana *Imprimir* se podrán ver los datos generales del dispositivo y si el usuario pulsa el botón "Opciones de impresión" aparecerán las opciones del dispositivo referentes a sus capacidades en la ventana *Opcs impresion*. El usuario puede modificar los valores por defecto de estas opciones y guardarlas pulsando el botón "Aceptar". La acción de ver la pantalla con las opciones de impresión es opcional. Si el usuario no solicitara ver dichas opciones se tomarían los valores por defecto contenidos en el fichero XML del dispositivo. El objeto *infoImpresion* es el encargado de almacenar el valor de estas opciones y una vez que termine, aparecerá de nuevo la ventana *Imprimir*. En esta ventana el usuario debe rellenar todos los datos necesarios para la impresión del documento, así como

proporcionar un fichero de texto plano o de tipo XML para completar los datos de la plantilla seleccionada. Cuando el usuario termina pulsa el botón *Imprimir* y el objeto *infoImpresion* recopila toda esta información y crea otro objeto *Cliente* para enviarle los datos al gestor de entradas. Éste decodifica la información y la reconoce como una petición de impresión. Este tipo de peticiones las gestiona el *motor de impresión*, pero antes el *gestor de la base de datos* almacena la información sobre el documento a imprimir: el nombre del documento y el contenido de los datos que completan la plantilla seleccionada. El *motor de impresión* solicita al *gestor de base de datos* todo lo que necesita para generar el fichero XML con la información de impresión del documento. Primero se obtiene toda la información relativa a la plantilla y al documento y se añade al fichero XML. Aunque en la figura no aparezca, si alguno de los datos es de tipo imagen el *gestor de imágenes* debe transformarlas al formato adecuado. A continuación se añaden también las opciones de impresión seleccionadas y por último se examinan las capacidades del dispositivo para ver si la impresión del documento es factible. En caso de no serlo, se buscarían alternativas para imprimir el documento con las capacidades del dispositivo disponibles. Al completar la creación del fichero XML, se realiza su transformación al lenguaje nativo del dispositivo mediante el *parser* y la hoja de estilo XSL correspondientes a dicho lenguaje. Al terminar todo el proceso, si el documento se obtiene correctamente se comunica al usuario el éxito de la operación a través de un mensaje informativo en una ventana emergente y se le muestra la opción de salir de la aplicación. Como se ve al final de la figura, cuando el usuario pulsa "Aceptar" aparece una ventana con la opción de salir de la aplicación y en ese momento termina su ejecución.

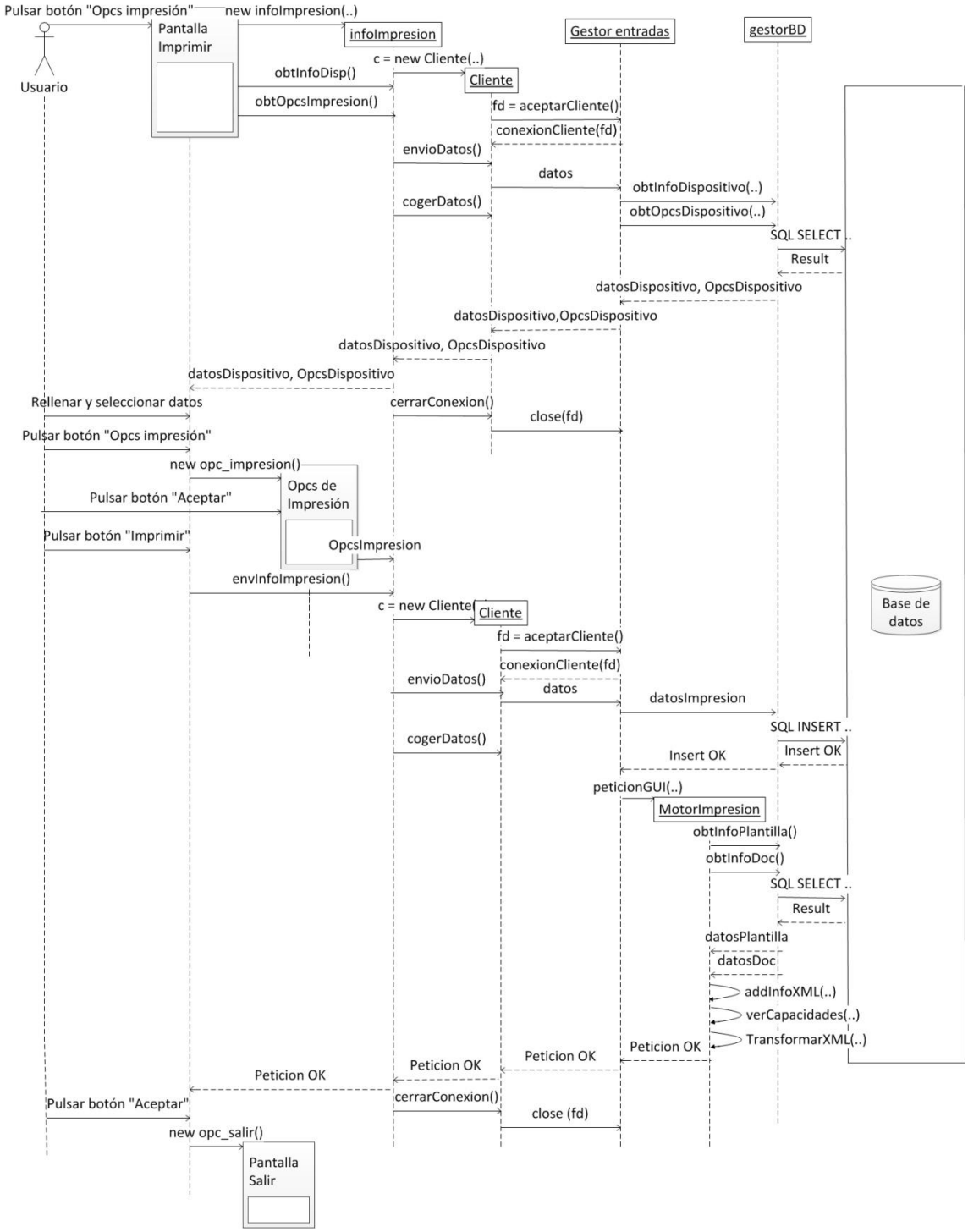

Figura C.8 Diagrama de secuencia de la impresión de documentos

Anexo D

Manual de usuario

Este anexo presenta el manual de usuario del sistema. Se explica el manejo de la interfaz que presenta la aplicación. La sección D.1 presenta los requisitos mínimos que permiten la utilización del sistema. La sección D.2 contiene las instrucciones a seguir por los usuarios, guiándolos por todas las opciones disponibles en las ventanas diseñadas. La sección D.3 incluye una guía para usar la las opciones disponibles para los administradores del sistema o de los dispositivos.

D.1 Requisitos mínimos de uso del sistema

Los requisitos para utilizar al sistema son los siguientes:

- Hardware: cualquier ordenador o dispositivo físico capaz de soportar la aplicación SILO. Deberá estar provisto de conexión a internet para realizar conexiones remotas.
- Software: la aplicación formará parte de uno de los módulos del producto SILO de HP. Por tanto, se necesita la instalación del producto SILO. Deberá funcionar en entornos cuyo sistema operativo soporte la máquina virtual java 1.6.0 o superior. El sistema deberá estar conectado a un gestor de bases de datos Oracle 10g Enterprise Edition Release 10.0.2.0 - 64bit Production, que será el utilizado por SILO.

D.2 Manual de usuario

D.2.1 Primeros pasos

Se inicia la aplicación y aparece un diálogo que solicita al usuario su identificación. Se debe introducir un nombre de usuario y una contraseña, que serán validados por el sistema. La figura D.1 muestra una captura de esta pantalla de inicio.

Figura D.1 Identificación de usuario

La gestión de los usuarios forma parte de la aplicación SILO y es el administrador del sistema el que proporciona nombre y contraseña a los usuarios. Todos los usuarios registrados se encuentran en la base de datos y es ahí donde se debe realizar su validación. Si los datos introducidos son correctos se da paso a la primera pantalla de la aplicación. En caso contrario, se mostrará el mensaje de error que aparece en la figura D.2.

Figura D.2 Mensaje de error en la identificación de usuario

La pantalla de inicio de la aplicación, que aparece en la figura D.3, muestra a la izquierda una lista con las plantillas del usuario, siempre que haya creado alguna previamente y la haya guardado. Si no tiene plantillas aparecerá vacía. En este caso la única opción disponible será ver las opciones de plantilla para crear una nueva. Se permiten realizar una serie de operaciones pulsando los distintos botones que aparecen:

- El botón "Opciones de plantilla" permite acceder a todas las opciones que se pueden realizar sobre las plantillas. Si el usuario no dispone de ninguna plantilla, se le permitirá crear una nueva. En cambio, si tiene alguna plantilla disponible podrá ver, editar o duplicar la plantilla que seleccione. La figura D.5 muestra estas opciones.

- El botón "Opciones de impresión" permite acceder a todas las opciones de impresión de las que dispone el dispositivo. Antes de pulsar este botón se debe seleccionar una de las plantillas disponibles. El dispositivo al que harán referencia las opciones de impresión será el que esté asignado a la plantilla. Como se puede ver en la figura D.3 se ha seleccionado una de las plantillas y en el cuadro de texto del

dispositivo se muestra su nombre. Las opciones de impresión que aparezcan en la siguiente pantalla dependerán del tipo dispositivo.

- El botón "Imprimir" es el que permite la impresión de los documentos. Antes de pulsarlo, además de seleccionar una plantilla, se deben rellenar previamente los datos correspondientes al documento o los documentos que se desean imprimir.

- El botón "Opciones de configuración" permite acceder a todas las opciones de configuración de las que dispone un dispositivo. Estas opciones sólo están disponibles para usuarios con permiso de administrador. Se solicitará al usuario que se identifique de nuevo, pero esta vez como administrador. El dispositivo se podrá seleccionar de una lista de dispositivos disponibles que se mostrará al usuario, una vez que se valide su identificación. Las opciones de configuración que se muestren dependerán del dispositivo elegido.

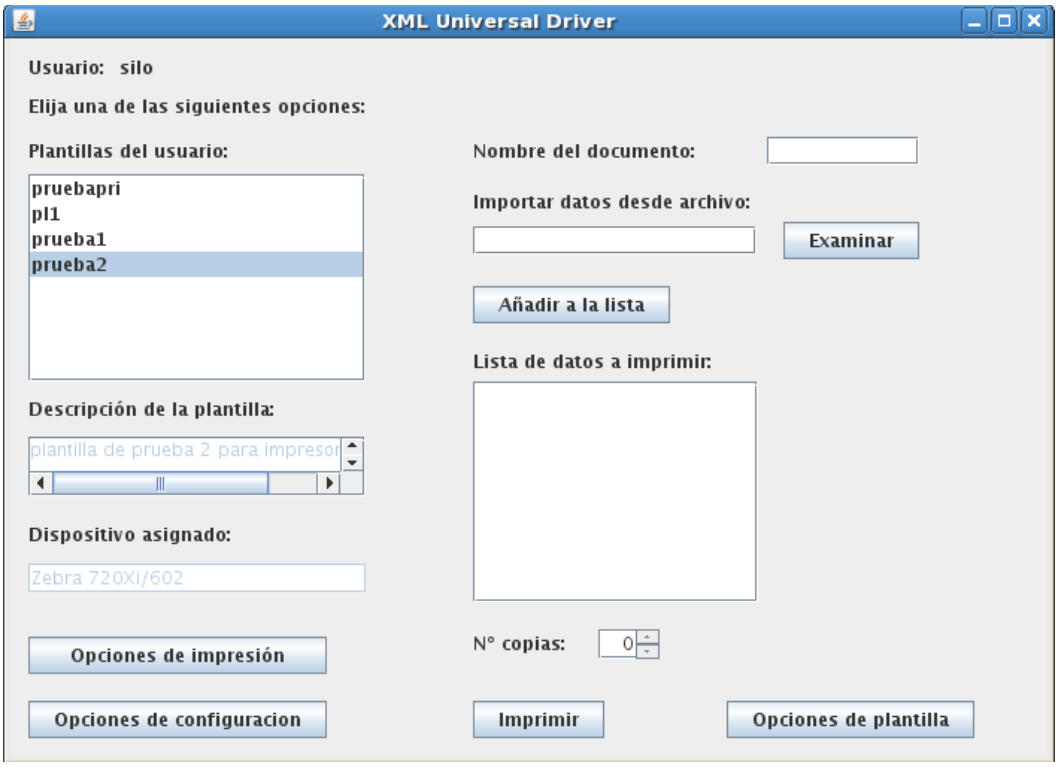

Figura D.3 Pantalla de inicio de la aplicación.

Si no se rellenan correctamente los datos de la pantalla de inicio, se mostrará el mensaje de error de la figura D.4.

Figura D.4 Mensaje de error al rellenar datos

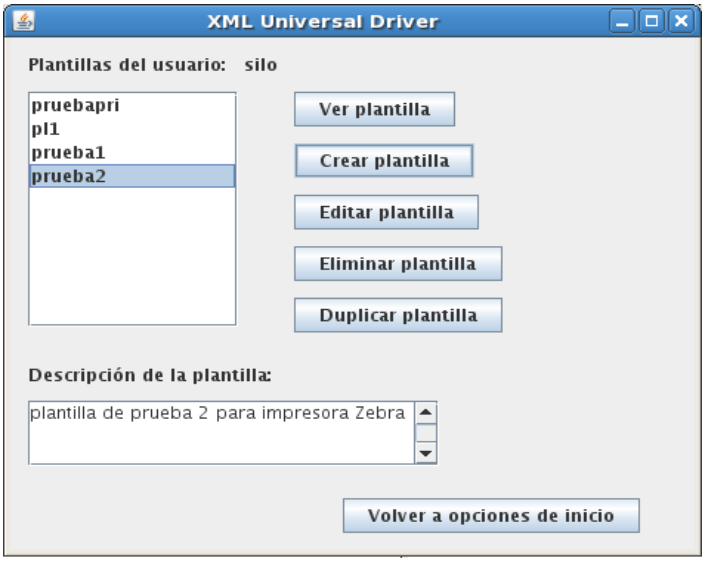

Figura D.5 Pantalla de opciones de plantilla

D.2.2. Creación de plantillas

Una de las opciones básicas de esta aplicación de impresión es la generación de plantillas. El usuario debe seguir los primeros pasos y comenzar la creación de una nueva plantilla pulsando el botón “Crear plantilla” de la figura D.5. Esto nos mostrará la pantalla de la figura D.6. En ella aparece un formulario donde introducir los datos generales de la plantilla.

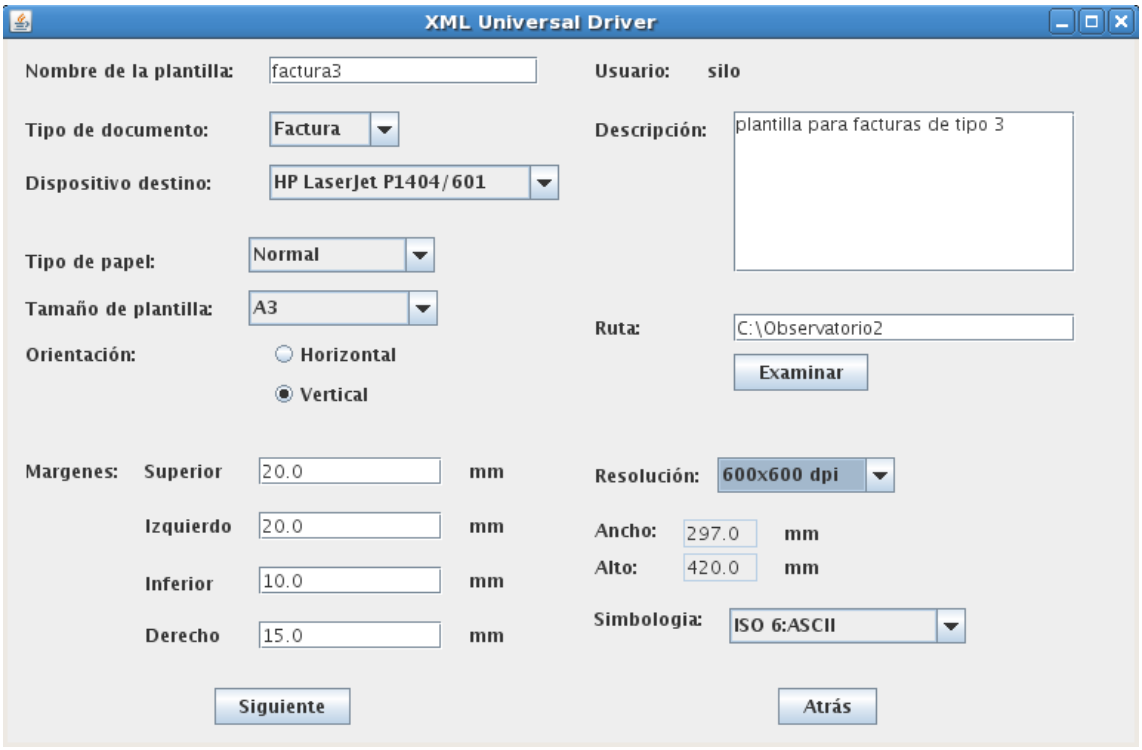

Figura D.6 Pantalla de inicio de creación de plantillas

El primer paso es rellenar el nombre de la plantilla. Si el usuario dispone de alguna plantilla no puede utilizar ninguno de sus nombres para la nueva plantilla. A continuación puede rellenar el campo que indica la descripción de la plantilla. Esta descripción debe ser breve y servirá para que el usuario reconozca fácilmente la plantilla una vez que esté creada. Otro de los campos a rellenar es la ruta donde se almacenarán los documentos de impresión que utilicen la plantilla. Esta ruta puede introducirse pulsando el botón "Examinar" y recorriendo los directorios de carpetas disponibles hasta seleccionar una de ellas. Se debe seleccionar el tipo de documento que corresponda a la plantilla que se va a generar. Los tipos disponibles son: albarán, factura, etiqueta e informe. A continuación se debe seleccionar uno de los dispositivos disponibles. En caso de que no se pueda obtener ningún dispositivo se mostrará el mensaje de error de la figura D.7 y se cerrará la aplicación.

El resto de opciones a rellenar dependen del dispositivo seleccionado. Se debe elegir un tipo de papel y un tamaño para la plantilla. El tamaño de la plantilla se corresponderá con uno de los tamaños de papel soportados por el dispositivo o si lo desea el usuario puede seleccionar un tamaño de tipo personalizado. En tal caso se deberán introducir las medidas del papel en los campos donde se indica su anchura y altura. Junto con las medidas del papel que se utilizará para la impresión, se debe definir su orientación, que puede ser vertical u horizontal. Si se cambia la orientación establecida, las medidas que hacen referencia a la altura y anchura se invertirán. Para completar las medidas de la plantilla se han de definir sus cuatro márgenes. Cada margen puede tener un valor distinto y el dato introducido debe ser un número real con un decimal separado por un

punto. Los márgenes delimitarán el área de la plantilla donde se situarán sus elementos. Por último se seleccionan las opciones restantes que hacen referencia a las capacidades del dispositivo. Se elegirá una de las resoluciones de impresión disponibles y un conjunto de símbolos. Si en cualquier momento se desea cancelar la creación de la plantilla se puede volver a la pantalla anterior pulsando el botón "Atrás" o salir de la aplicación cerrando la ventana. Al terminar de rellenar todos estos datos se pulsará el botón "Siguiente" para continuar con la creación de los elementos de la plantilla. La figura D.8 muestra esta pantalla donde aparecen todas las opciones relativas a los elementos.

Figura D.7 Mensaje de error de dispositivos

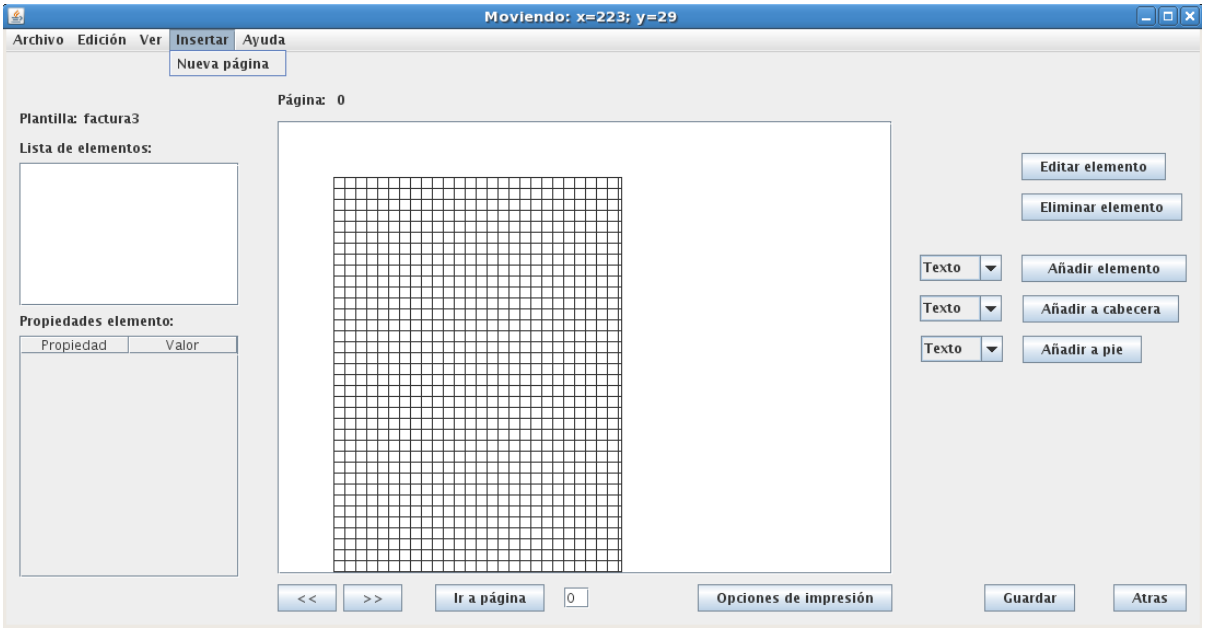

Figura D.8 Pantalla de creación de elementos de la plantilla

La nueva ventana contiene un menú superior que permite añadir páginas, crear, editar o eliminar elementos y guardar la plantilla. El centro de la pantalla contiene un panel con una figura dividida por celdas que representa la plantilla. La zona de la izquierda presenta una lista, que contendrá los elementos de la plantilla que se vayan añadiendo a la página actual. La zona situada a la derecha contiene las opciones que permiten la creación de los elementos de la plantilla. La zona inferior de la pantalla permite

visualizar diferentes páginas de la plantilla, guardar la configuración de la plantilla creada y solicitar la impresión de un documento a partir de esta plantilla.

El primer paso, antes de crear un nuevo elemento para la plantilla, es añadir una página. Para ello, tal y como muestra la figura D.8, se debe seleccionar del menú superior la opción de insertar una nueva página. Si se intenta crear algún elemento sin haber insertado ninguna página se muestra el mensaje de error de la figura D.9.

Figura D.9 Mensaje de error en la creación de un nuevo elemento.

Una vez que la plantilla contiene al menos una página, ya se pueden añadir nuevos elementos. Para añadir un nuevo elemento se debe seleccionar primero el tipo de elemento que se desea crear y a continuación se pulsará uno de los botones disponibles, que permiten elegir la zona de la página donde se va a situar: cabecera, cuerpo o pie. Antes de introducir los datos correspondientes al elemento que se desea crear aparece una nueva ventana que permite establecer la posición de inicio del elemento. La figura D.10 muestra esta pantalla, donde se debe seleccionar, en primer lugar, el tipo de posición: absoluta o relativa. Una posición absoluta es la que establece las coordenadas en los ejes X e Y tomando como referencia la esquina superior izquierda delimitada por sus márgenes. Si se selecciona la posición relativa, se toma como referencia la posición final de uno de los elementos que contiene la página. La posición final de un elemento se encuentra en la esquina inferior derecha del área que ocupa. Este tipo de posición relativa no está disponible cuando no se ha creado ningún elemento previamente. Las posiciones introducidas deben tener como unidad de medida el milímetro. Si la posición es de tipo absoluto sólo se permite introducir datos enteros positivos, incluido el cero también. En cambio, si la posición es relativa se permite introducir datos enteros negativos para desplazar la posición hacia la izquierda del elemento tomado como referencia.

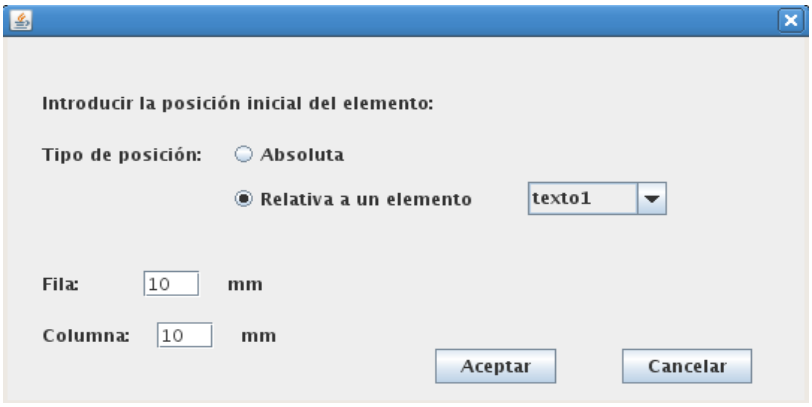

Figura D.10 Pantalla de creación de elementos de la plantilla

Se comprueba que la posición elegida se encuentra dentro del área de impresión de la página y que no colisiona con otros elementos. En caso de producirse un error al seleccionar la posición se mostrará la pantalla de la figura D.11 y se podrá volver a introducir una nueva posición.

Figura D.11 Mensaje de error de selección de posición de un nuevo elemento

Una vez que se admite la posición elegida para el nuevo elemento, aparecerá la pantalla correspondiente al tipo de elemento seleccionado.

D.2.3 Creación de un elemento de tipo texto

En la pantalla de la figura D.8 se debe seleccionar la opción de añadir un elemento de tipo texto. Una vez seleccionada su posición de inicio, se muestra la pantalla de la figura D.12 con las propiedades del elemento de texto a rellenar.

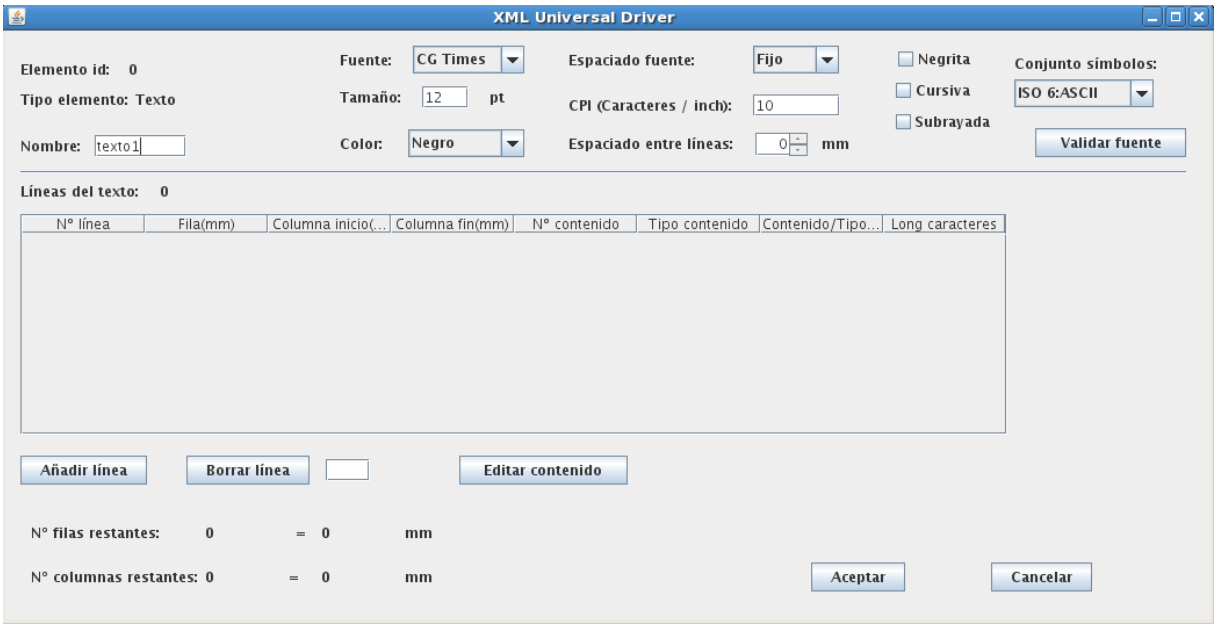

Figura D.12 Pantalla de creación de un elemento de tipo texto

La parte superior de la pantalla contiene una serie de campos que deben ser completados antes de añadir cualquier línea al elemento de texto. En primer lugar se debe asignar un nombre único al elemento para diferenciarlo del resto y a continuación se van rellenando el resto de campos. Los tipos de fuentes y conjuntos de símbolos se corresponden con los soportados por el dispositivo de impresión asignado a la plantilla. El tamaño de la fuente vendrá determinado por los valores introducidos en los campos *Tamaño* y *CPI*. El *Tamaño* corresponde a la altura medida en *pt*, que es la medida de fuente que usan las herramientas de edición de texto. El *CPI* corresponde a la longitud del dato, ya que mide el número de caracteres que se imprimirán por pulgada. Teniendo en cuenta que una pulgada son 25,4 milímetros, se calcularía el número de caracteres por milímetro, ya que es la medida utilizada para las posiciones. El espaciado de caracteres debe ser fijo cuando se crea una plantilla para ser impresa. Si fuera variable no se podría calcular con precisión la longitud del texto. Si se añaden varias líneas al elemento, se puede introducir un espaciado adicional entre ellas. Si el valor del espacio es cero, se aplicará el valor mínimo predeterminado. Se pueden aplicar al elemento opciones para modificar su aspecto, aumentando la densidad de sus caracteres, inclinándolos o incluso subrayándolos. Por último se da la opción de elegir el conjunto de símbolos. Si se desea mantener el que se seleccionó al comienzo de la creación de la plantilla, se deberá elegir la opción *Defecto*. Una vez completados todos los datos acerca de las propiedades del elemento, se debe pulsar el botón “Validar fuente”. Si se pulsa este botón sin haber rellenado todos los datos, aparecerá el mensaje de error de la figura D.13.

Figura D.13 Mensaje de error de falta de propiedades del texto

Si los datos introducidos son correctos se pueden añadir nuevas líneas de texto al elemento. Para ello se debe pulsar el botón "Añadir línea" cada vez que se quiera crear una nueva. La pantalla de la figura D.14 es un diálogo donde se muestra la información necesaria para crear o editar líneas.

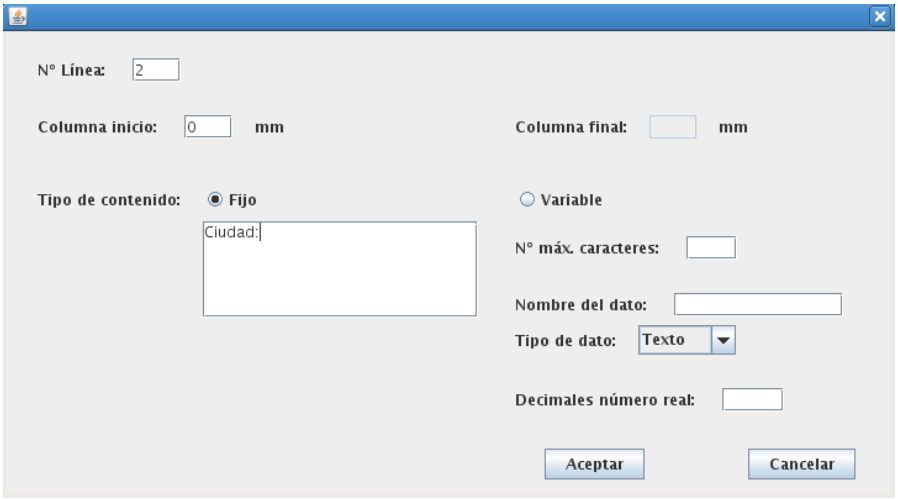

Figura D.14 Pantalla de creación de una línea de texto

Cada línea se identifica con un número y la primera línea debe tener el número cero. La columna de inicio de la línea toma como referencia la distancia a la columna de inicio del elemento. El valor mínimo permitido es el cero y el máximo la distancia entre la columna de inicio del elemento de texto y el margen derecho de la página. A continuación se debe definir el tipo de contenido: fijo o variable. Si el contenido es fijo se debe introducir su valor tal y como se muestra en la figura D.14. Estos elementos de texto aparecerán siempre en los documentos que se impriman a partir de la plantilla. Si se selecciona la opción de contenido variable se deben rellenar el resto de opciones acerca de su contenido. Estos contenidos de tipo variable se usan cuando se desea que el texto incluido en una posición concreta del documento a imprimir no sea siempre el mismo. Los datos que deben completar la información de estos contenidos variables son la longitud máxima de caracteres que van a tener, un nombre único que permita identificarlos y el tipo de dato que van a representar: texto, entero, real con decimales, fecha o fecha con hora. La forma de rellenar estos datos se puede ver en la figura D.15.

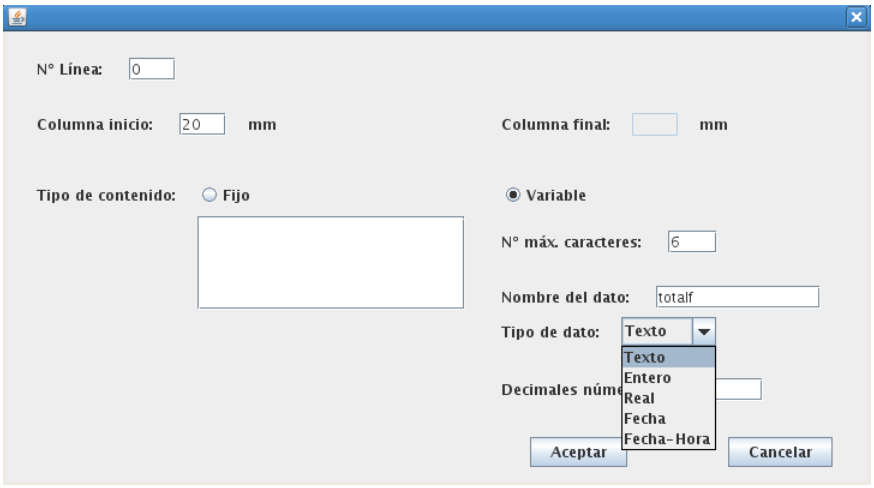

Figura D.15 Pantalla de creación de una línea de texto

Se pueden añadir varias líneas con el mismo número siempre que la columna de inicio sea superior a la longitud total de la línea. Esto permite tener varios *contenidos* de texto en una misma línea. La figura D.16 muestra un ejemplo de un elemento de texto que contiene varias líneas y contenidos en cada una de ellas. La tabla que aparece presenta un resumen de la información de cada contenido de cada línea. A medida que se van creando líneas van apareciendo en esta tabla y la información que ésta contiene sirve de ayuda para posicionar y crear nuevas líneas y contenidos. Otra información de interés es la que aparece en la zona inferior de la pantalla. El número de filas y columnas restantes proporcionan una idea de la longitud y altura máxima que puede alcanzar en total el elemento de texto. Esta información aparecerá en todas las pantallas de creación de elementos, que se explican en las siguientes secciones.

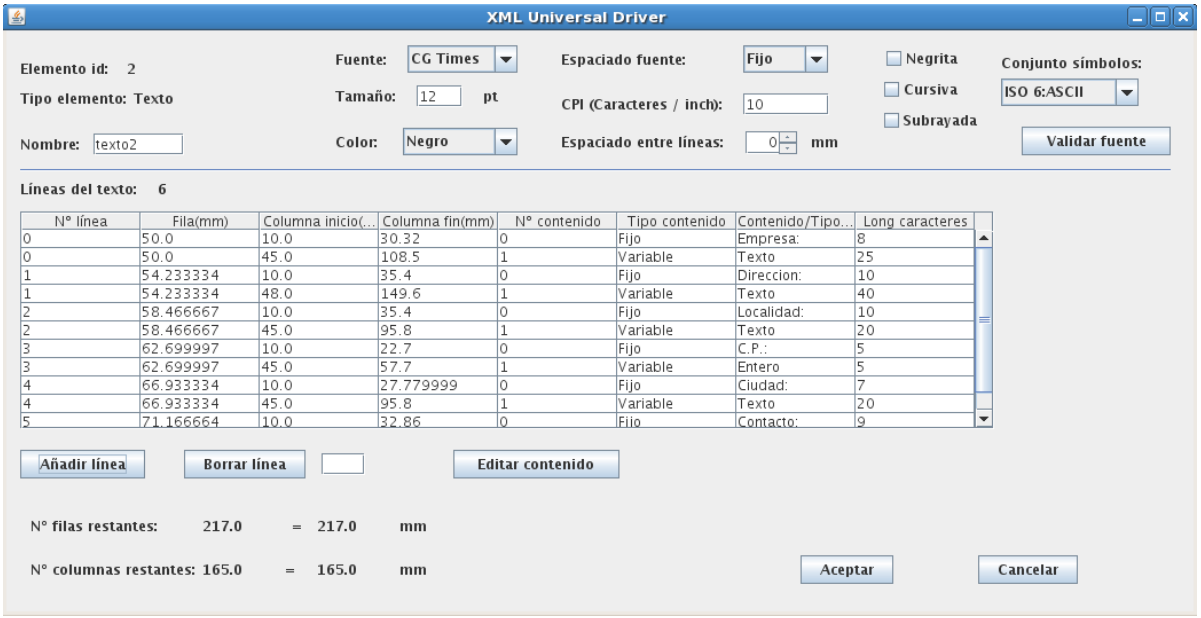

Figura D.16 Pantalla de creación de una línea de texto

No es necesario que los elementos de texto se estructuren en líneas y con varios elementos de texto diferentes dentro de ellas. Se pueden crear elementos con una única línea y un único dato. La opción de estructurar estos elementos resulta útil cuando se quiera introducir texto como en un formulario, con una serie de campos y valores separados por varias columnas y alineados.

Si se ha cometido algún error al crear alguna de las líneas del elemento, se podrán borrar o incluso se podrá editar su contenido de nuevo. Una vez que se termine de configurar el elemento, se pulsará "Aceptar" y será guardado como un nuevo elemento de la página actual. En cualquier momento se puede anular la creación del elemento pulsando el botón "Cancelar". Al pulsar cualquiera de estos dos botones se regresa a la pantalla de la figura D.8.

D.2.4 Creación de un elemento de tipo imagen

En la pantalla de la figura D.8 se debe seleccionar la opción de añadir un elemento de tipo imagen. Una vez seleccionada su posición de inicio, se muestra la pantalla de la figura D.17 con las propiedades del elemento de tipo imagen a rellenar.

En primer lugar se debe introducir un nombre único que identifique al elemento dentro de la página. A continuación se introduce la localización del archivo que contiene la imagen, cuyo formato debe ser *jpeg* o *bitmap*. Si no se conoce la ruta exacta se puede acceder a la organización de directorios y archivos, a través del botón "Examinar", y elegir uno de los archivos de imagen. El resto de campos a rellenar son las medidas de la imagen y el número de *píxels* que contiene por filas y columnas. °). La información acerca de los píxeles se puede obtener de las propiedades del archivo de la imagen.También se puede rotar la imagen un número de grados determinado (90°, 180° o 270).

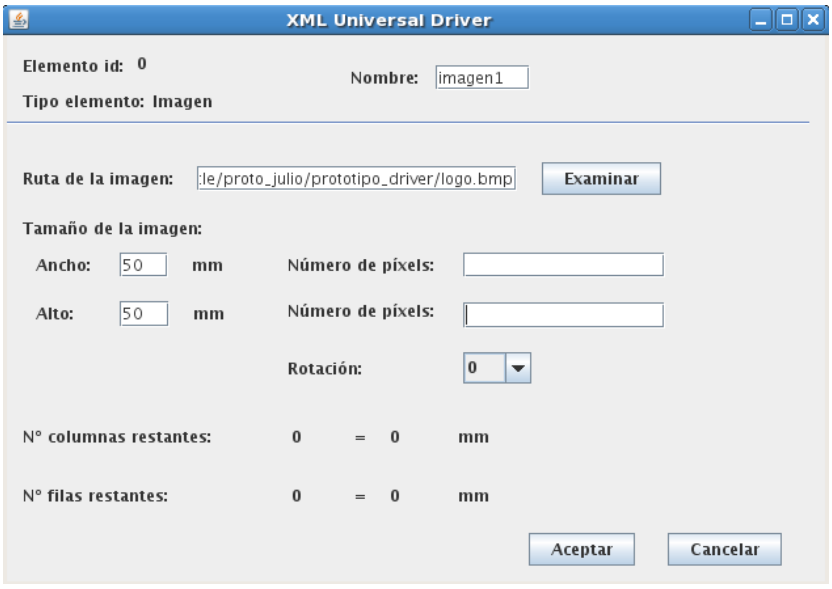

Figura D.17 Pantalla de creación de un elemento de tipo imagen

Una vez que se termine de configurar el elemento, se pulsará “Aceptar” y será guardado como un nuevo elemento de la página actual. En cualquier momento se puede anular la creación del elemento pulsando el botón “Cancelar”. Al pulsar cualquiera de estos dos botones se regresa a la pantalla de la figura D.8.

D.2.5 Creación de un elemento de tipo línea

En la pantalla de la figura D.8 se debe seleccionar la opción de añadir un elemento de tipo línea. Una vez seleccionada su posición de inicio, se muestra la pantalla de la figura D.18 con las propiedades del elemento de tipo línea a rellenar.

En primer lugar se debe introducir un nombre único que identifique al elemento dentro de la página. A continuación se debe elegir cuál será la orientación que tendrá la línea: horizontal o vertical. El resto de campos a rellenar son la longitud del elemento y el grosor de la línea. Ambos valores deben introducirse en milímetros. La longitud de la línea debe ser mayor que cero y menor que la distancia entre la posición de inicio del elemento y el margen derecho. El grosor de la línea debe ser como mínimo de 0,3 milímetros. El valor máximo dependerá del soportado por la impresora. El color de la línea generalmente será negro, pero se podrá seleccionar otro en caso de que la impresora lo soporte.

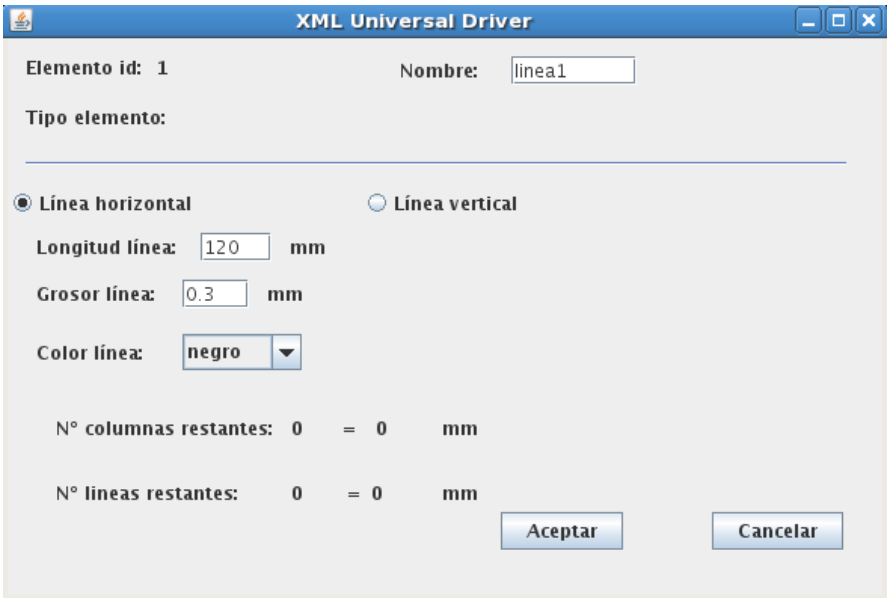

Figura D18. Pantalla de creación de un elemento de tipo línea

Una vez que se termine de configurar el elemento, se pulsará “Aceptar” y será guardado como un nuevo elemento de la página actual. En cualquier momento se puede anular la

creación del elemento pulsando el botón “Cancelar”. Al pulsar cualquiera de estos dos botones se regresa a la pantalla de la figura D.8.

D.2.6 Creación de un elemento de tipo caja

En la pantalla de la figura D.8 se debe seleccionar la opción de añadir un elemento de tipo caja. Este tipo de elemento no solicita la definición de su posición de inicio y muestra directamente la pantalla de la figura D.19 con las propiedades del elemento de tipo caja a rellenar.

En primer lugar se debe introducir un nombre único que identifique al elemento dentro de la página. Entre las opciones que aparecen se debe seleccionar primero el tipo de posición del elemento, que puede ser al igual que en el resto de elementos, absoluta o relativa. Si se elige el tipo de posición absoluta se debe introducir la posición de inicio del elemento compuesta por una fila y una columna de inicio. A continuación se debe introducir tanto su longitud en el eje vertical como en el eje horizontal. Si se desea situar la caja rodeando alguno de los elementos pertenecientes a la página actual se deberá elegir una posición relativa y seleccionar uno de los elementos disponibles. Debajo de la lista de elementos aparecen cuatro campos donde introducir la distancia a la que se situará cada una de las cuatro líneas que forman la caja. Por último se rellenan el resto de opciones que corresponden al grosor, el color de las líneas y al color de fondo, que hace referencia al área de la caja delimitada por sus cuatro líneas. Todos los valores que hacen referencia a medidas y posiciones deben ser introducidos en milímetros.

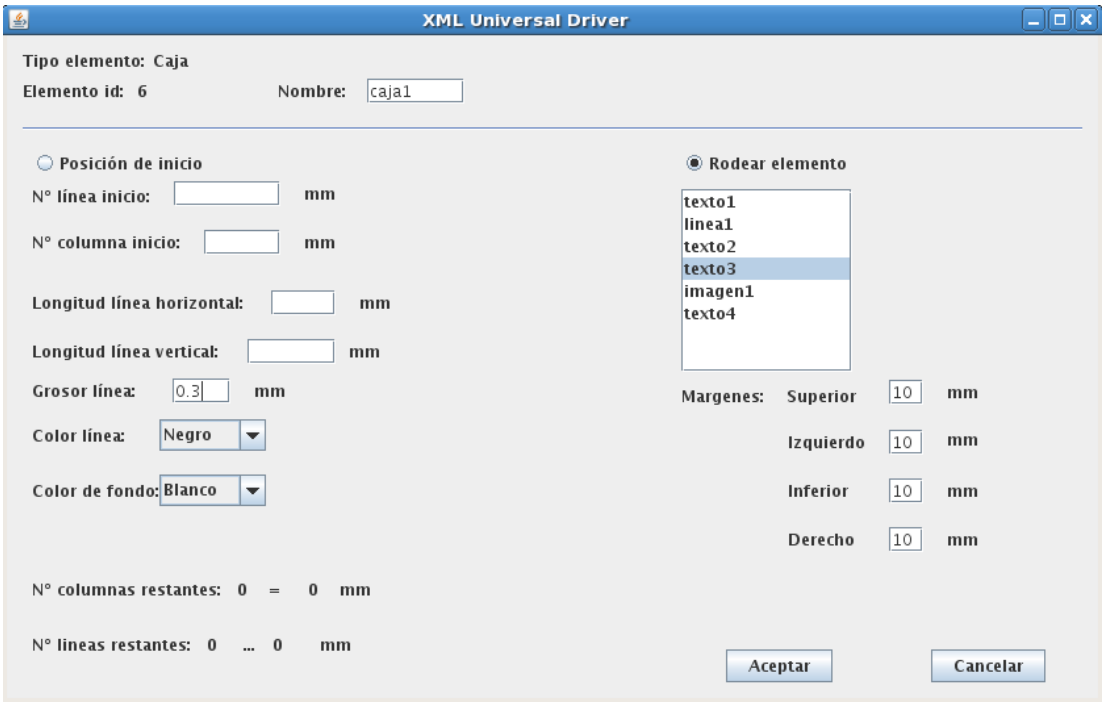

Figura D.19 Pantalla de creación de un elemento de tipo caja

Una vez que se termine de configurar el elemento, se pulsará “Aceptar” y será guardado como un nuevo elemento de la página actual. En cualquier momento se puede anular la creación del elemento pulsando el botón “Cancelar”. Al pulsar cualquiera de estos dos botones se regresa a la pantalla de la figura D.8.

D.2.7 Creación de un elemento de tipo código de barras

En la pantalla de la figura D.8 se debe seleccionar la opción de añadir un elemento de tipo código de barras. Una vez seleccionada su posición de inicio, se muestra la pantalla con las propiedades del elemento de tipo código de barras a rellenar. La figura D.20 muestra las opciones para crear códigos de barras para etiquetadoras Zebra y la figura D.21 muestra un mayor número de opciones a completar para etiquetadoras Printronix e impresoras que no soporten códigos de barras.

En ambas figuras los primeros pasos a seguir son los mismos. En primer lugar se debe introducir un nombre único que identifique al elemento dentro de la página. Entre las opciones que aparecen se debe seleccionar primero el tipo de código de barras para que permanezcan habilitadas sólo aquellas opciones aplicables al código seleccionado. La dimensión del código de barras es un campo que muestra de forma automática el tipo de dimensión del código al seleccionarlo (1D o 2D).

Primero se van a explicar todas las opciones que aparecen en la figura D.20. En esta figura se puede ver como se ha seleccionado el código de barras de tipo *EAN-13* y se han deshabilitado muchas de las opciones que no son aplicables a este tipo de códigos. Para comprender mejor estas opciones se van a explicar las características de los códigos de barras. Un código de barras es un conjunto de líneas y espacios, que siguen un patrón predefinido de forma estándar. Están formados por los siguientes elementos:

- Barra: elemento oscuro.
- Espacio: elemento blanco o claro.

Los códigos de barras se clasifican en varias categorías:

- Según el tipo de datos que acompañan al código:
    - Código alfanumérico: es el que permite representar caracteres alfabéticos y numéricos. El Código 39 es de este tipo.
    - Código numérico: es el que sólo permite representar números.
- Según la disposición de sus espacios:
    - Código discreto: es un código en el que los espacios entre caracteres no son representativos y pueden variar su anchura dentro de una tolerancia muy grande.
    - Código continuo: es un código en el que los espacios forman parte del mismo.

Independientemente del tipo de código de barras seleccionado, la altura y anchura definirán el área que vaya a ocupar el código en la plantilla. La anchura corresponde a la longitud del primer carácter al último carácter. La altura de barras corresponderá a la longitud del elemento oscuro o claro. El módulo corresponde a la barra o el espacio más estrecho del código. Las barras con espacios más anchos deben ser múltiplos al módulo. El ratio es la relación de multiplicidad de una barra o espacio respecto al módulo. Los caracteres de inicio y fin se colocan al principio y al final del código para indicar el sentido correcto de lectura. El dígito de *check* sirve para la comprobación del código y ver que es correcto. Los códigos pueden llevar datos asociados, numéricos o alfanuméricos, y se pueden situar encima o debajo del conjunto de barras y espacios. Para ello se debe especificar la impresión de líneas de interpretación tal y como aparece en la figura D.20. Además se pueden añadir varios símbolos adicionales a los datos mediante las opciones de número de símbolos y número total de símbolos. Estos símbolos suelen ser de 2 a 5 caracteres. La detección de error permite seleccionar el grado de corrección que se aplicará al código en caso de que no sea correcto. El factor de magnificación proporciona varios tamaños para la impresión del código. El modo establece la forma de imprimir los espacios entre los caracteres de los datos del código. Se puede seleccionar el modo automático para que se analicen los datos sin necesidad de seleccionar un modo manual. La truncación es una opción que permite eliminar la parte derecha del código que soporte esta opción. Los valores específicos para cada campo no se detallan porque se considera que el usuario está familiarizado con los mismos.

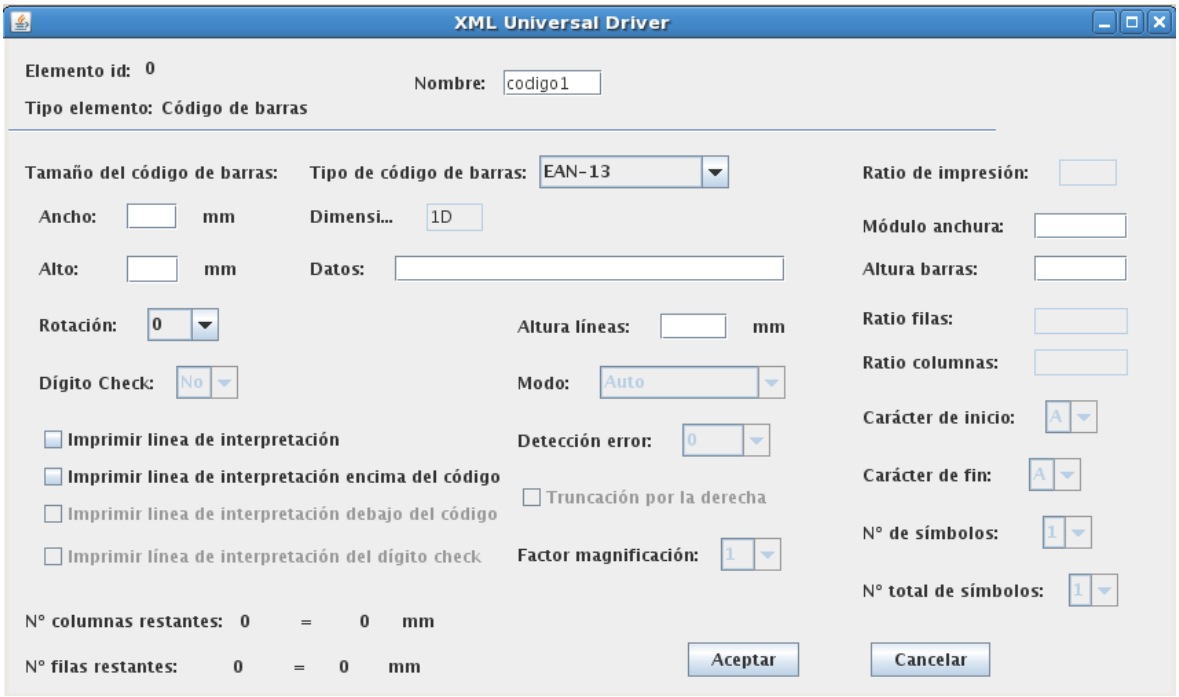

Figura D.20 Pantalla de creación de un elemento de tipo código de barras para etiquetadoras Zebra

Ahora se van a explicar todas las opciones que aparecen en la figura D.21. En esta figura se puede ver como se ha seleccionado el código de barras de tipo *128A*. Muchas de las opciones que aparecen en la figura D.21 son las mismas que ya se han explicado para los código de la figura D.20. Las opciones separadas por una barra mostrarán los valores adecuados según el código seleccionado. Se supone que el usuario está familiarizado con las propiedades de cada código de barras y no es necesario explicar cuáles corresponden a cada código. En esta figura D.21 aparecen nuevas opciones como la posibilidad de remarcado de líneas o el acortamiento de las barras centrales del código. La opción *Tipo de código industrial* se usará para crear códigos con formatos que se ajusten a los que se usan en la industria . La opción *INIT* es opcional y se usa en los códigos de tipo *Aztec* para especificarlos como *inicializadores* al ser leídos. La opción *KIX* es un parámetro opcional para los códigos de tipo *RoyalMail* que indica que se debe seguir ese tipo de formato. La opción *BIN* se usa en códigos de tipo *Micro-PDF417* para que codifique sus datos haciendo una compactación en bytes. La opción *Zipper* se emplea para códigos postales. La opción *FCC* se usa para definir el formato del código y su tamaño y la opción *INFO* es la que define el formato del campo *datos*. Si no se introduce ningún valor se toman por defecto los valores 11 y 1 para *FCC* e *INFO*, respectivamente. Estas opciones hacen referencia al código *Australian 4-State*. Las opciones *Format X* y *Format Y*, junto con las opciones *Append X* y *Append Y*, se usan para definir el formato de los código de barras de tipo *Aztec*. *Format* se usa para definir el formato del código de barras, estableciendo el número de niveles y la longitud máxima del campo de datos del código. Acepta valores que van desde 0 a 192. *Append* es un parámetro opcional que especifica, en el caso de X el número de símbolos que se entrelazan unos con otros, y en el caso de Y el número de símbolos que se pueden añadir al final del campo de datos. Comprende valores numéricos que van de 1 a 26. Los valores de las opciones correspondientes a los *ratios* suelen ser un número real entre 0.3 y 0.9 pulgadas. Los valores que se deben introducir en estos campos deben ser enteros que van de 3 a 99. Las opciones *Dimensión X* e *Y* establecen la anchura y la altura, respectivamente, de los elementos más estrechos del código en base al tamaño del tipo de punto seleccionado (*IGP* / Impresora). Por último, hay unas opciones para los códigos de tipo *QR*. Primero se debe seleccionar si se van a concatenar los datos particionados lo componen. En caso de seleccionar la concatenación se podrán introducir los valores adecuados acerca de la longitud de las particiones y el número total de particiones a concatenar. El valor a introducir debe ser entre 1 y 16. Otro dato necesario para la opción de concatenación es el byte de paridad, que se debe introducir en hexadecimal.

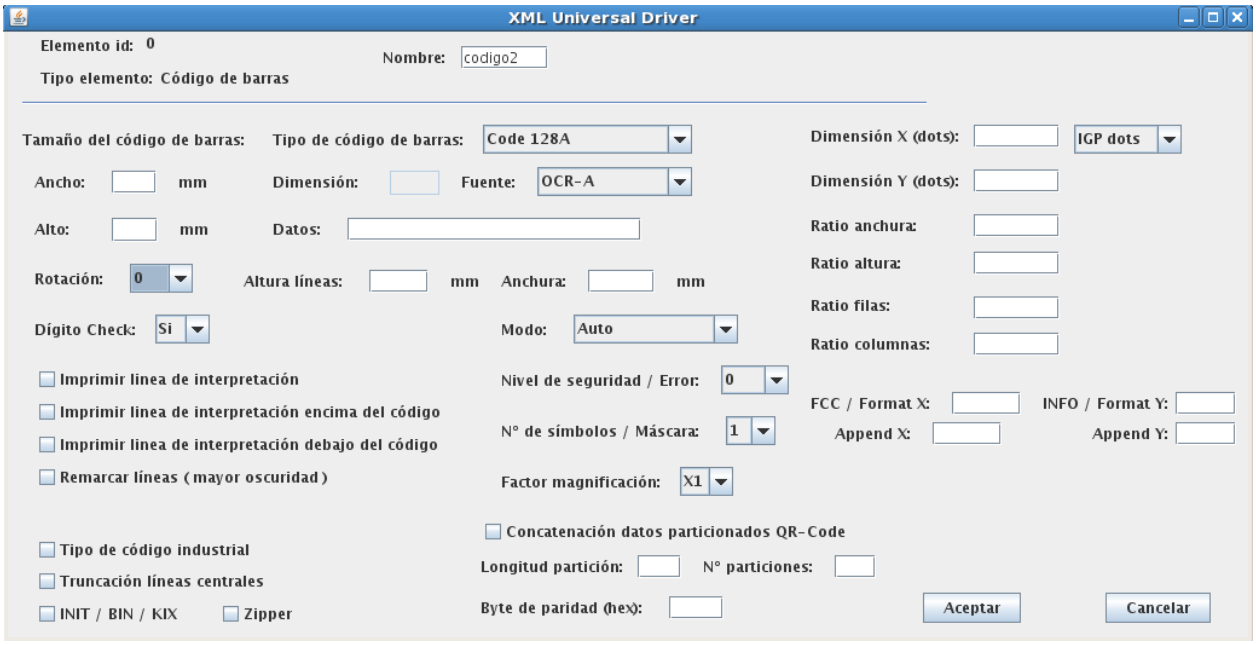

Figura D.21 Pantalla de creación de un elemento de tipo código de barras para etiquetadoras Printronix

Una vez que se termine de configurar el elemento, se pulsará “Aceptar” y será guardado como un nuevo elemento de la página actual. En cualquier momento se puede anular la creación del elemento pulsando el botón “Cancelar”. Al pulsar cualquiera de estos dos botones se regresa a la pantalla de la figura D.8.

D.2.8 Guardar una plantilla

A medida que se van creando elementos y añadiéndolos a la plantilla, van apareciendo en una lista, tal y como muestra la figura D.22. Si se selecciona un elemento se muestra una lista con sus propiedades en forma de tabla y se dibuja el área que ocupa dentro de la página con un cuadrado rojo.

Una vez que se hayan creado todos elementos que van a formar parte de la plantilla, si se desea guardar su configuración, se debe pulsar en el botón “Guardar” o seleccionar en el menú superior “Archivo” la opción de “Guardar”. Si la plantilla es guardada con éxito se muestra el mensaje de la figura D.23. En caso de que se abandone la aplicación, cerrando la ventana, antes de guardar la plantilla, se perderán los datos.

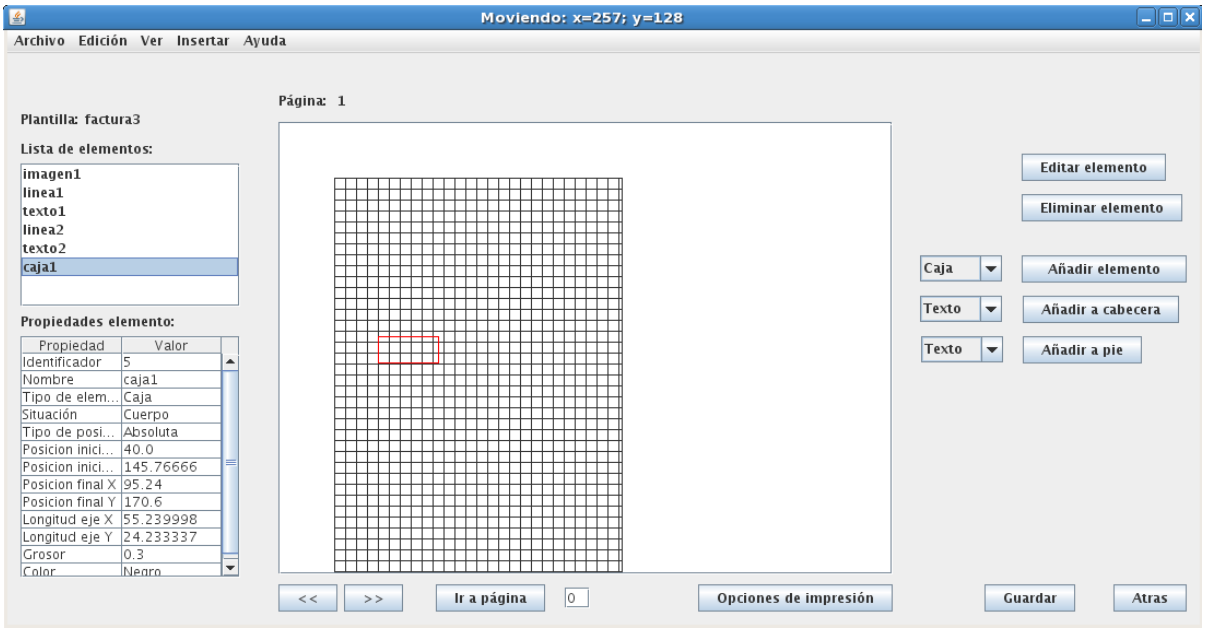

Figura D.22 Pantalla de creación de elementos

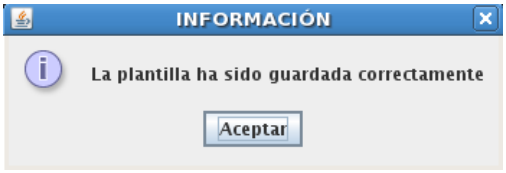

Figura D.23 Mensaje de información sobre la plantilla guardada

D.2.9 Impresión de documentos

En la pantalla de la figura D.3 se mostraba el inicio de la aplicación. Esta pantalla permite la impresión de documentos, pero no es la única forma de hacerlo. La otra forma en la que se pueden imprimir documentos es a partir de la pantalla de la figura D.25. En esta sección se explica el funcionamiento de ambas opciones.

La figura D.24 muestra la misma pantalla que se veía en la figura D.3, pero con sus campos completos con la información necesaria para la impresión de un documento. El primer paso es seleccionar una de las plantillas disponibles. Si no hubiera ninguna plantilla no se permitirá la impresión de documentos. Al seleccionar una de las plantillas se muestra su descripción y el dispositivo que tiene asignado. A partir de este momento se podrán seleccionar las opciones de impresión correspondientes al dispositivo de la plantilla, pulsando el botón "Opciones de impresión". Las instrucciones acerca de las distintas opciones de impresión del dispositivo seleccionado se describen en la sección D.2.9.

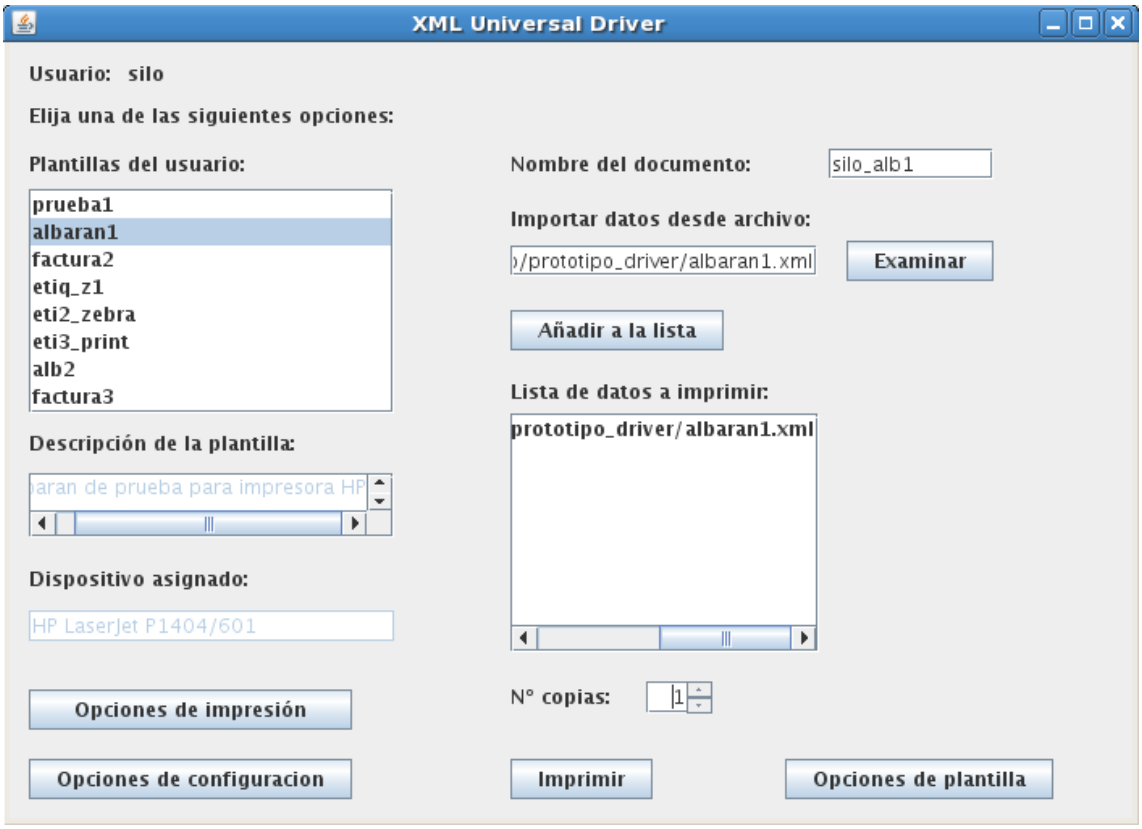

Figura D.24 Pantalla de inicio para la impresión de documentos

Para la impresión del documento se deben rellenar todos los campos que presenta la pantalla de la figura D.24. Esto implica introducir el nombre del documento y seleccionar el número de copias. Por último se debe introducir, al menos, un fichero de datos que sirva para complementar la información de la plantilla seleccionada. El botón "Examinar" permite buscar en la rama de directorios del ordenador el fichero de datos, tal y como se muestra en la figura D.25. El formato de este fichero debe ser de texto plano o de tipo XML. La extensión del fichero de texto debe ser ".txt" y debe contener los datos variables de fueron definidos en la plantilla al crearla. Estos datos deben separarse por comas y estar ordenados por orden de aparición en el documento, siguiendo un orden de líneas de menor a mayor y dentro de cada línea por columnas, de izquierda a derecha. La extensión del fichero XML debe ser ".xml" y debe contener un nodo raíz, llamado "datos", que a su vez contenga una lista de etiquetas o nodos. Cada una de estas etiquetas deberá corresponder a un tipo de dato variable de la plantilla. La etiqueta será el nombre del dato y deberá contener un elemento, que será el texto asignado al dato. Tras seleccionar un fichero de datos que cumpla con cualquiera de estos dos formatos, se debe pulsar el botón "Añadir a la lista". Si es correcta su extensión se añadirá a la lista de datos a imprimir. Si el dispositivo de la plantilla seleccionada permite la impresión de varios documentos, se podrán añadir tantos ficheros de datos como se desee a esta lista.

Figura D.25 Pantalla de búsqueda de archivos

Una vez que se hayan introducido todos los datos y el archivo necesario para imprimir el documento, se pueden seleccionar las opciones de impresión. En caso de que no se modifiquen las opciones de impresión del dispositivo, se tomará su configuración por defecto. Para obtener el documento a imprimir se pulsará el botón "Imprimir". Si todos los datos son correctos, se creará el archivo de impresión del documento y se guardará en la ruta especificada al crear la plantilla. A continuación se mostrará una ventana que le preguntará si desea abandonar la aplicación. En caso que pulse "Sí" se cerrarán todas las ventanas y si pulsa "No" permanecerá en la pantalla de inicio.

La otra forma de imprimir documentos se puede realizar al terminar de crear una plantilla. En la figura D.22 se mostraba la creación de una plantilla. Una vez que se guarde la plantilla, se puede imprimir a continuación un documento con esa misma plantilla. Para ello se debe pulsar el botón "Opciones de impresión" y aparecerá la pantalla de la figura D.26. Guarda cierta similitud con la pantalla de la figura D.24. Esta pantalla muestra el nombre del dispositivo asignado a la plantilla creada, junto con su estado y localización. Se muestra además una lista de dispositivos para poder cambiar el dispositivo si se desea, pero para ello las capacidades del nuevo dispositivo deberán ser compatibles con las del antiguo. También se permite seleccionar otra plantilla de la lista de plantillas que hay disponible para ese usuario. Se debe rellenar el nombre del documento e introducir archivos de datos que completen la plantilla. El formato de estos archivos es el mismo que se explicó para la figura D.24. Se pueden añadir y eliminar documentos de la lista si se desea rectificar. Por último también se pueden seleccionar las opciones de impresión pulsando el botón "Ver opciones de impresión". Estas opciones se explican en la siguiente sección. Al pulsar el botón "Imprimir" se generará el documento final listo para ser impreso.

Una vez obtenido el documento de impresión, se le pregunta si desea salir de la aplicación. La figura D.27 muestra este diálogo. En caso de que su repuesta sea "Sí" se cerrará la aplicación. Si no desea abandonar la aplicación se volverá a la pantalla de inicio de la figura D.24.

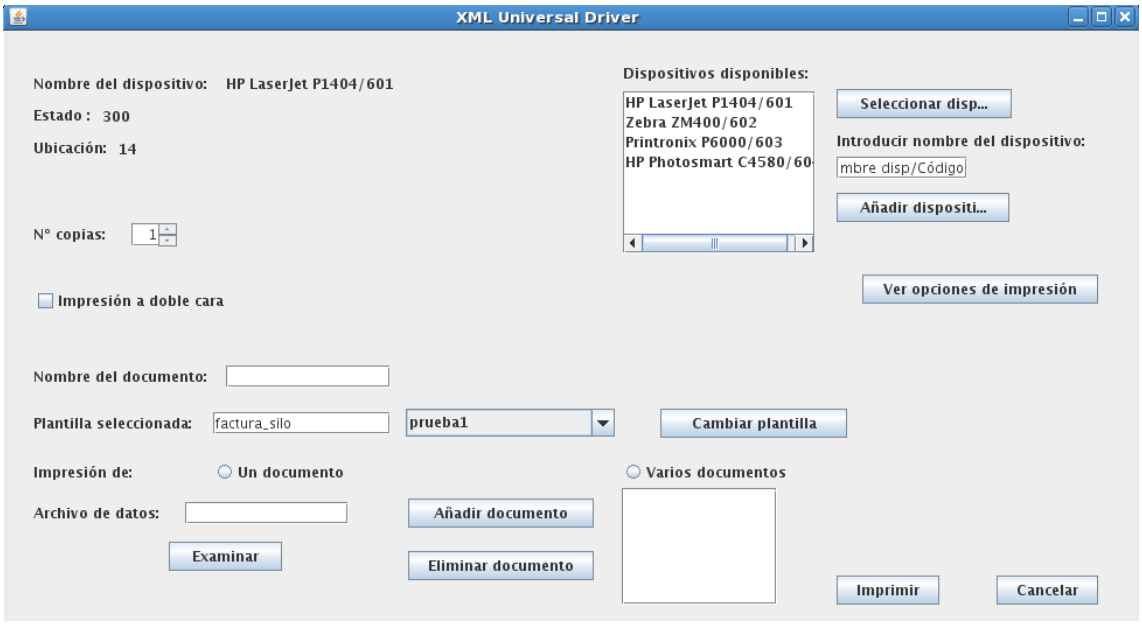

Figura D.26 Pantalla de impresión de documentos

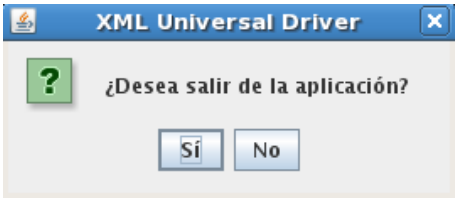

Figura D.27 Opción de salir de la aplicación

D.2.10 Opciones de impresión

Esta sección describe las opciones de impresión correspondientes a los distintos fabricantes de los dispositivos de impresión que soporta la aplicación. Primero se describen las opciones para etiquetadoras, *Zebra* y *Printronix*, seguidas de las opciones para impresoras *HP*.

La figura D.28 muestra las opciones de impresión para una etiquetadora *Zebra*. Las opciones acerca del tipo de papel, tamaño, orientación y resolución no se pueden modificar, al haber sido configuradas previamente durante la creación de la plantilla. El resto de opciones se pueden consultar en los manuales proporcionados por el fabricante para obtener más detalles. El tipo de impresión para las etiquetadoras hace referencia a la forma en que se impregna el tóner en el papel. El modo de impresión permite configurar el movimiento sobre el papel: si se rebobina, avanza o se corta. Las velocidades de impresión y de desplazamiento se miden en pulgadas por segundo y su valor suele ser 2. Se permite la rotación de la etiqueta a 180° y el desplazamiento del

margen superior izquierdo. Por último se permite modificar el grado de intensidad de la impresión, remarcando más o menos el texto.

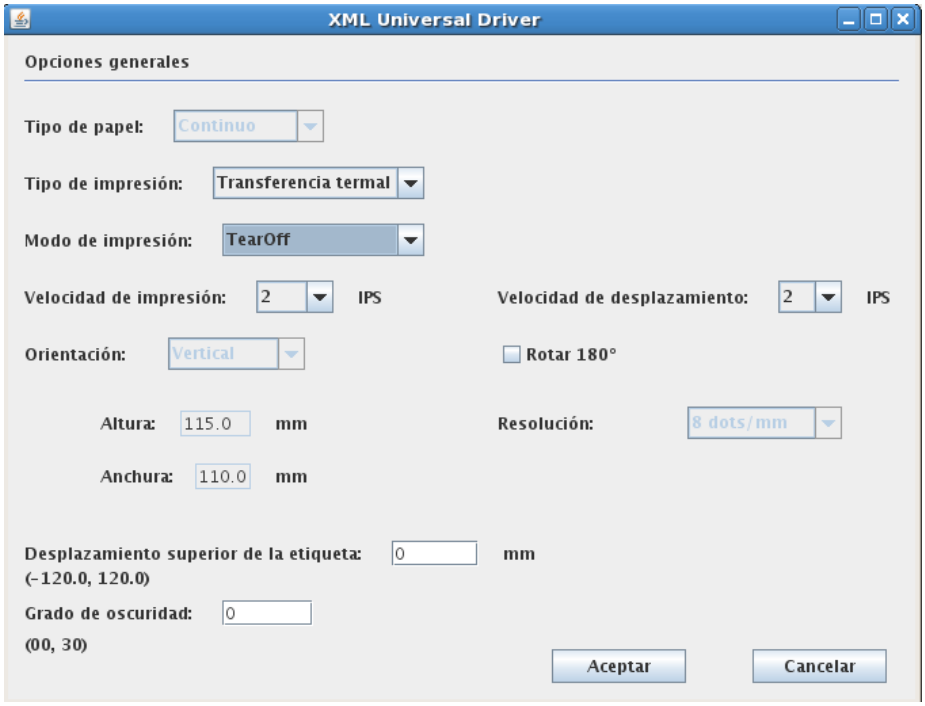

Figura D.28 Pantalla de opciones de impresión para etiquetadoras Zebra

Las opciones de impresión de las etiquetadoras *Printronix* son muy similares a las de *Zebra* y se muestran en la figura D.29. La única diferencia es que no se permite desplazar el margen superior izquierdo de la etiqueta y que el modo de impresión hace referencia al tipo de etiquetadora.

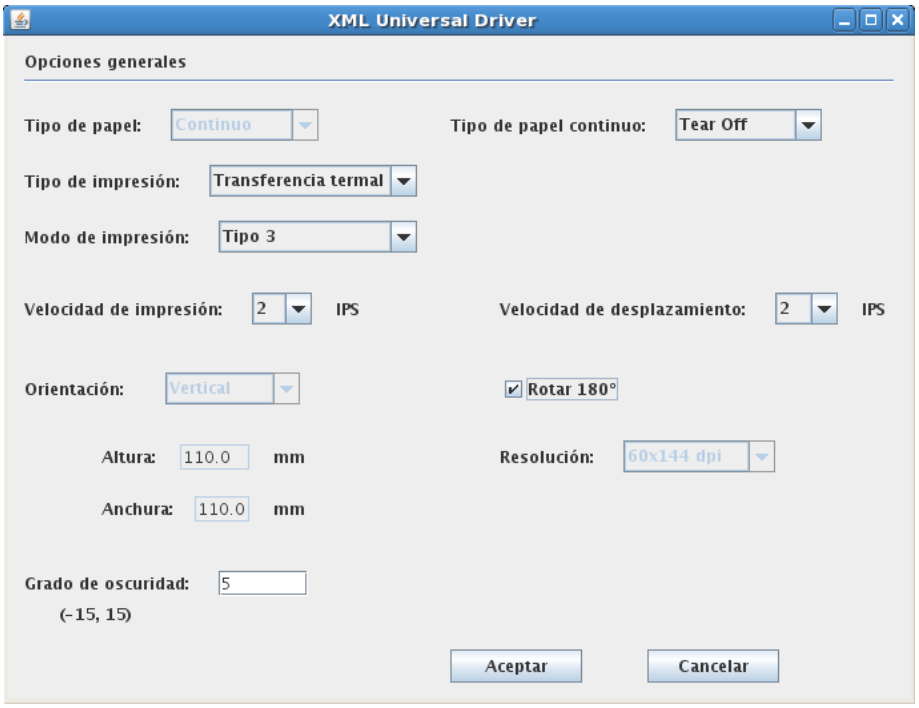

Figura D.29 Pantalla de opciones de impresión para etiquetadoras Printronix

Una vez que se hayan seleccionado todos los valores deseados se debe pulsar el botón "Aceptar" para que se guarden los nuevos valores. Si se pulsa "Cancelar" se tomarán los últimos valores guardados o los que había por defecto. En cualquier caso, se volverá a la pantalla de la figura D.24.

Las figuras D.30, D.31, D.32, D.33 y D.34 muestran las opciones de impresión para una impresora *HP*. Se dividen en cinco categorías según su utilidad: generales, de ajuste de tamaño, de ahorro, de color y especiales. Las opciones básicas acerca del tipo de papel, tamaño, orientación y resolución no se pueden modificar, al haber sido configuradas previamente durante la creación de la plantilla. El resto de opciones se pueden consultar en los manuales proporcionados por el fabricante para obtener más detalles.

La figura D.30 presenta una serie de opciones básicas en las que se pueden seleccionar las bandejas de entrada y salida, así como el alimentador de hojas de forma automática. Se podrá seleccionar el número de páginas a imprimir o si se desea imprimir todas. Por lo general se imprimirán todas las páginas, pero si se quisieran seleccionar unas pocas se introducirán siguiendo el formato especificado. Otra opción será la impresión de las páginas ordenadas de forma normal o en orden inverso.

La figura D.31 presenta las opciones correspondientes al ajuste de tamaño. Se puede elegir entre mantener el tamaño real de la plantilla o ajustarla a un nuevo tamaño de página, siempre que sea posible. Otra opción es la aparición de bordes en las páginas.

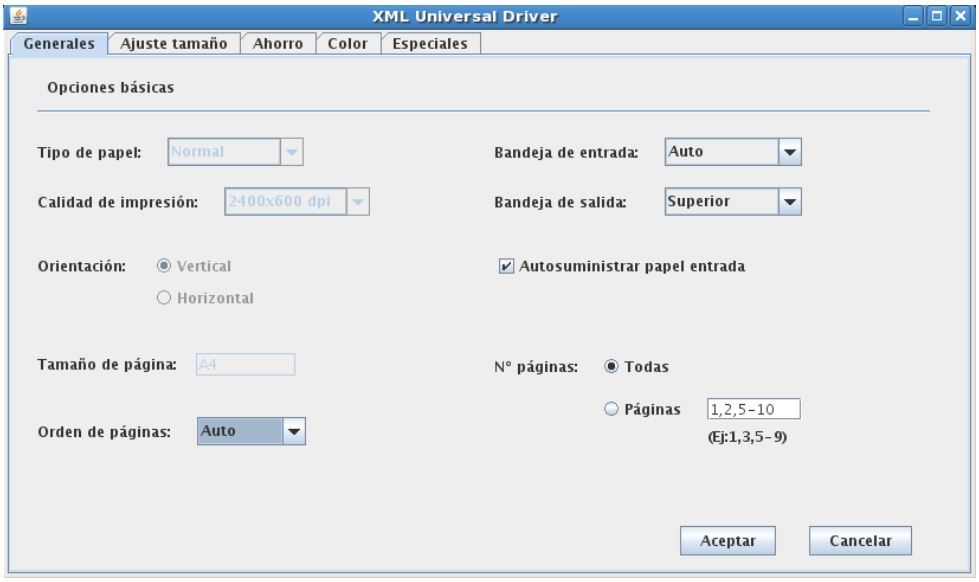

Figura D.30 Pantalla de opciones generales para impresoras HP.

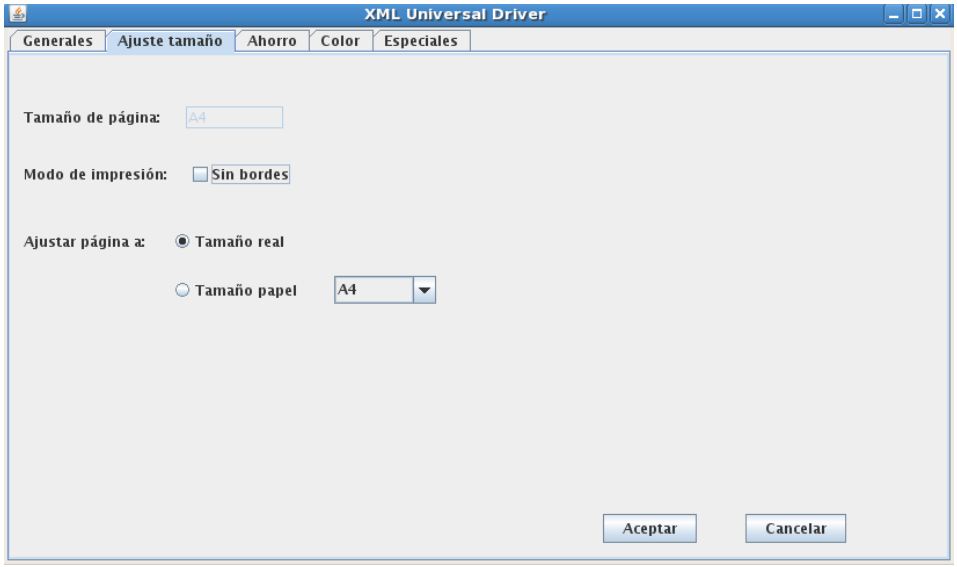

Figura D.31 Pantalla de opciones de ajuste de tamaño para impresoras HP

La figura D.34 presenta las opciones de ahorro. Estas opciones permiten la impresión a doble cara y de varias páginas por hoja, el encuadernado y el grapado de documentos. Si se selecciona un número de páginas por hoja se debe establecer un orden para las páginas. El modo de ahorro de tóner es una opción que sirve para disminuir el consumo del mismo haciendo que perdure su uso.

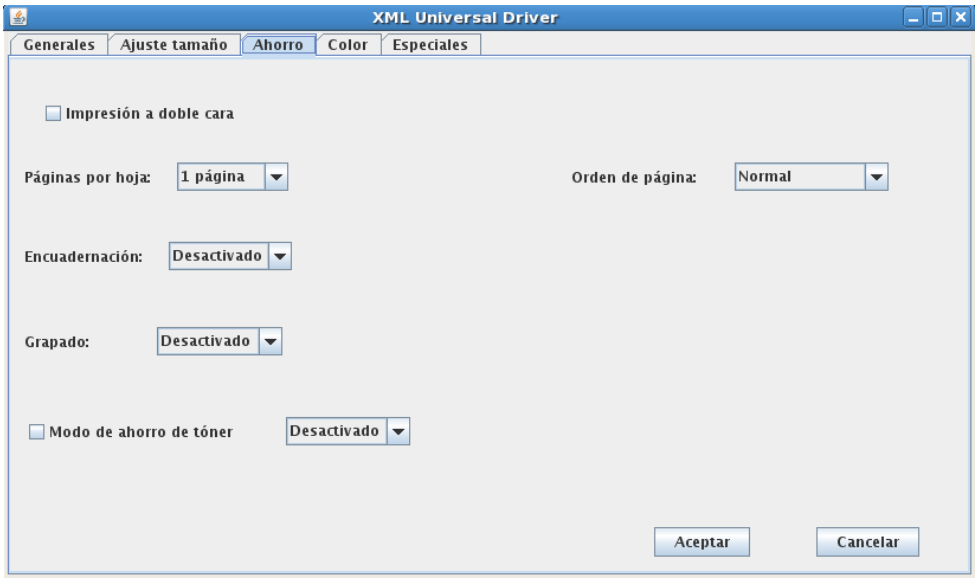

Figura D.32 Pantalla de opciones de ahorro para impresoras HP

La figura D.33 presenta las opciones de color. Estas opciones estarán disponibles siempre que el dispositivo pueda imprimir en color. Permiten seleccionar la mezcla de colores a usar y la impresión en colores grises. El resto de opciones de colores forman parte de las opciones de configuración.

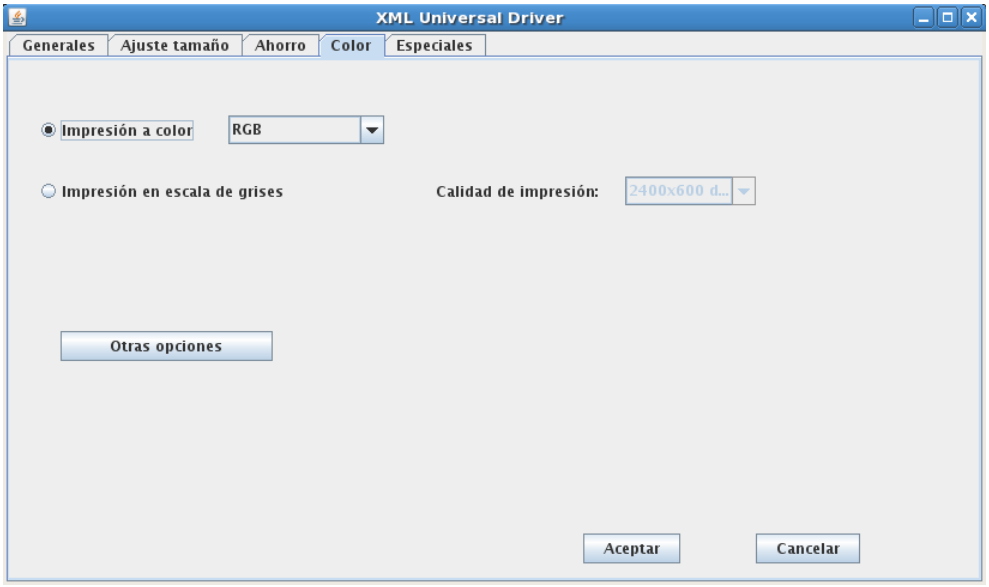

Figura D.33 Pantalla de opciones de color para impresoras HP

La figura D.34 presenta las opciones especiales de impresión. Estas opciones permiten la impresión de todo el documento en negro, aunque presente colores. Se puede también

activar la configuración máxima de impresión de páginas por minuto y seleccionar el volumen de tinta a usar.

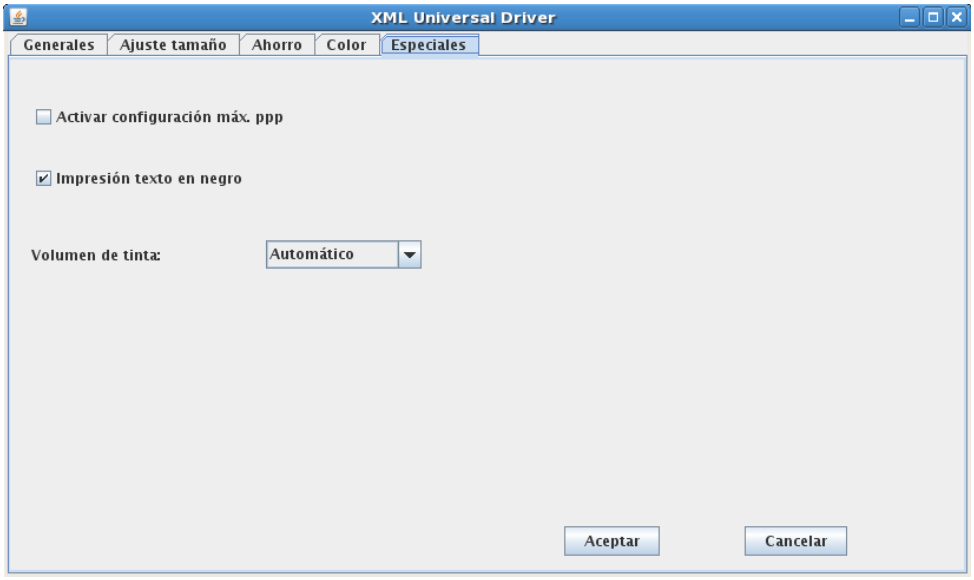

Figura D.34 Pantalla de opciones especiales para impresoras HP

Una vez configuradas todas estas opciones se puede pulsar el botón “Aceptar” desde cualquiera de las pestañas de opciones para que se guarden los nuevos valores. Si se pulsa “Cancelar” se tomarán los últimos valores guardados o los que había por defecto. En cualquier caso, se volverá a la pantalla de la figura D.24.

D.3 Manual de administrador

Esta sección es una guía para los usuarios que tengan permiso para administrar y configurar los dispositivos de impresión disponibles.

D.3.1 Primeros pasos

Se inicia la aplicación y aparece un diálogo que solicita al usuario su identificación. Se debe introducir un nombre de usuario y una contraseña, que serán validados por el sistema. La figura D.35 muestra una captura de esta pantalla de inicio.

Figura D.35 Identificación de usuario

La gestión de los usuarios forma parte de la aplicación SILO y es el administrador del sistema el que proporciona nombre y contraseña a los usuarios. Todos los usuarios registrados se encuentran en la base de datos y es ahí donde se debe realizar su validación. Si los datos introducidos son correctos se da paso a la primera pantalla de la aplicación. En caso contrario, se mostrará el mensaje de error que aparece en la figura D.36.

Figura D.36 Mensaje de error en la identificación de usuario

La pantalla de inicio de la aplicación, que aparece en la figura D.3, muestra a la izquierda una lista con las plantillas del usuario, siempre que haya creado alguna previamente y la haya guardado. Si no tiene plantillas aparecerá vacía. En este caso la única opción disponible será ver las opciones de plantilla para crear una nueva. Se permiten realizar una serie de operaciones pulsando los distintos botones que aparecen:

- El botón “Opciones de plantilla” permite acceder a todas las opciones que se pueden realizar sobre las plantillas. Si el usuario no dispone de ninguna plantilla, se le permitirá crear una nueva. En cambio, si tiene alguna plantilla disponible podrá ver, editar o duplicar la plantilla que seleccione. La figura D.5 muestra estas opciones.

- El botón “Opciones de impresión” permite acceder a todas las opciones de impresión de las que dispone el dispositivo. Antes de pulsar este botón se debe seleccionar una de las plantillas disponibles. El dispositivo al que harán referencia las opciones de impresión será el que esté asignado a la plantilla. Como se puede ver en la figura D.37 se ha seleccionado una de las plantillas y en el cuadro de texto del dispositivo se muestra su nombre. Las opciones de impresión que aparezcan en la siguiente pantalla dependerán del tipo dispositivo.

- El botón “Imprimir” es el que permite la impresión de los documentos. Antes de pulsarlo, además de seleccionar una plantilla, se deben rellenar previamente los datos correspondientes al documento o los documentos que se desean imprimir.

- El botón “Opciones de configuración” permite acceder a todas las opciones de configuración de las que dispone un dispositivo. Estas opciones sólo están disponibles para usuarios con permiso de administrador. Se solicitará al usuario que se identifique de nuevo, pero esta vez como administrador. El dispositivo se podrá seleccionar de una lista de dispositivos disponibles que se mostrará al usuario, una vez que se valide su identificación. Las opciones de configuración que se muestren dependerán del dispositivo elegido.

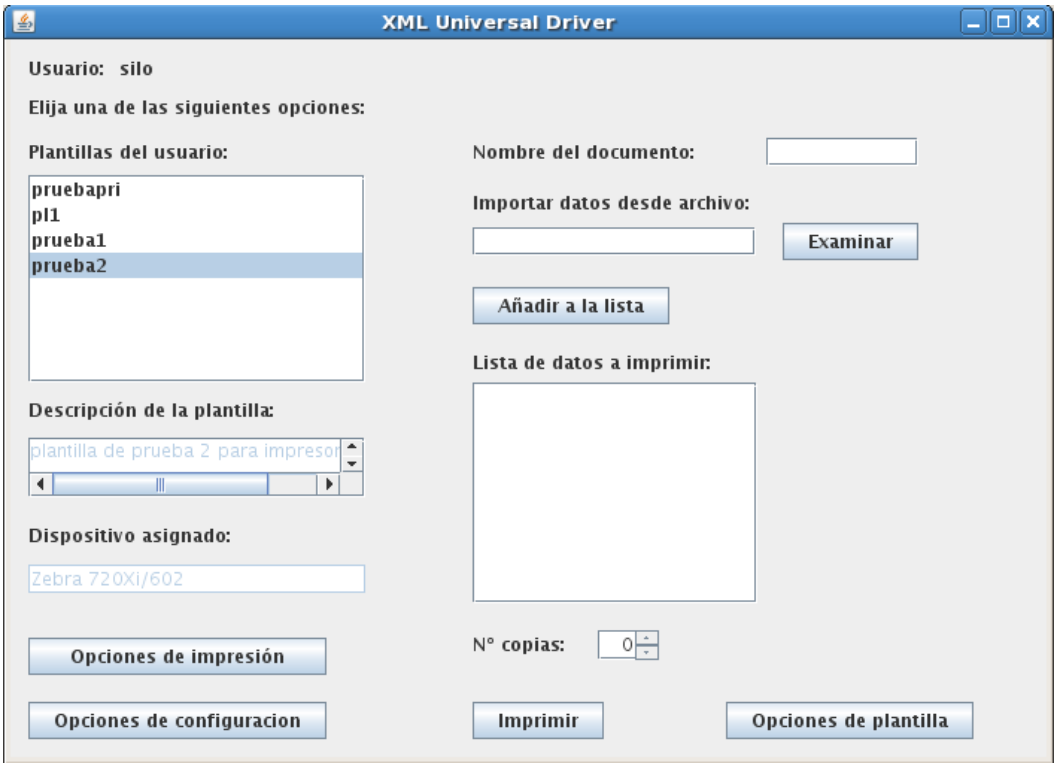

Figura D.37 Pantalla de inicio de la aplicación

D.3.2 Opciones de configuración

Partimos de que el usuario haya abierto la aplicación, se haya identificado y se encuentre en la pantalla de inicio de la figura D.37. El primer paso para acceder a la configuración de dispositivos es pulsar el botón “Opciones de configuración”. En ese momento, aparece el diálogo de la figura D.38 solicitando al usuario un nombre y contraseña que lo identifiquen como administrador del sistema. Si los datos son correctos, se mostrará la pantalla que aparece en la figura D.39. En esta pantalla se muestra una lista con todos los dispositivos disponibles para ser configurados. Se debe

seleccionar uno de ellos y pulsar “Siguiente” para comenzar la configuración del dispositivo. Si se pulsa el botón sin haber seleccionado ninguno de ellos se mostrará el error de la figura D.40. Para cancelar la configuración de dispositivos y volver a las opciones de inicio de la figura D.37 se debe pulsar ”Volver a inicio”.

Figura D.38 Pantalla de identificación del administrador de dispositivos

Figura D.39 Pantalla que muestra la lista de dispositivos configurables

Figura D.40 Mensaje de error de selección de dispositivo.

Primero se van a explicar las opciones de configuración para etiquetadoras, tanto *Zebra* como *Printronix*. La figura D.41 muestra estas opciones, que pueden ser consultadas en

los manuales proporcionados por el fabricante para obtener más detalles. El tipo de impresión para las etiquetadoras hace referencia a la forma en que se impregna el tóner en el papel. El modo de impresión permite configurar el movimiento sobre el papel: si se rebobina, avanza o se corta. Si se elige la opción de cortar (*Tear Off*) se puede ajustar la distancia a la que se producirá el corte, tomando como referencia el final de la etiqueta. La velocidad de impresión se mide en pulgadas por segundo y su valor por defecto suele ser 2. El aumento de este valor puede ser significativo si se imprimen muchas etiquetas al mismo tiempo. Se permite ajustar el desplazamiento del margen superior izquierdo y el grado de intensidad con el que se imprimirá, que permite oscurecer o aclarar el texto. La opción modo protegido se usa para evitar que se puedan modificar las opciones de impresión manualmente desde el panel de botones de la etiquetadora. Si se desea que se vuelva a imprimir en caso de que se produzca algún error, se deberá activar la opción correspondiente. Se permite modificar los prefijos correspondientes a las instrucciones de control y de formato. En caso de cambiar estos prefijos se deberá introducir el valor del símbolo elegido en hexadecimal. Se puede seleccionar el tipo de calibrado a utilizar. El botón "Calibrado" permite acceder a más opciones acerca del calibrado, tal y como muestra la pantalla que aparece en la figura D.42. Estas opciones permiten ajustar la medida de calibrado para cada sensor de forma específica. Una vez que se hayan seleccionado estas medidas se podrán guardar, pulsando el botón "Aceptar", o bien preservar los valores antiguos si se pulsa el botón "Cancelar". En ambos casos se volverá a la pantalla de la figura D.41.

Por último, se da la opción de guardar la configuración actual o bien de restablecer los valores de fábrica. Antes de aceptar la nueva configuración se puede imprimir una página de prueba pulsando el botón "Imprimir página de prueba". Si la nueva configuración es aceptable se pulsará el botón "Aceptar" para guardar la nueva configuración. Se podrán modificar y testear estas opciones tantas veces como se quiera imprimiendo diferentes páginas de prueba con cada configuración. Si se pulsa el botón "Cancelar" se conservará la configuración que existía previamente.

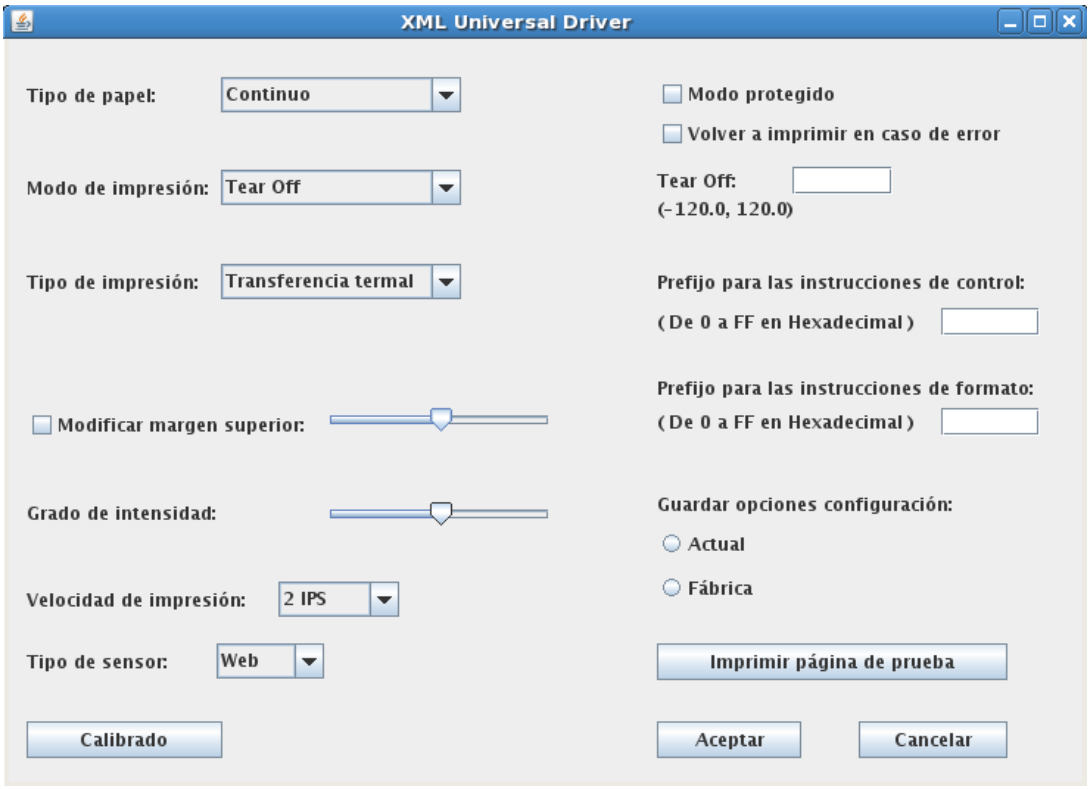

Figura D.41 Pantalla de opciones de configuración para etiquetadoras

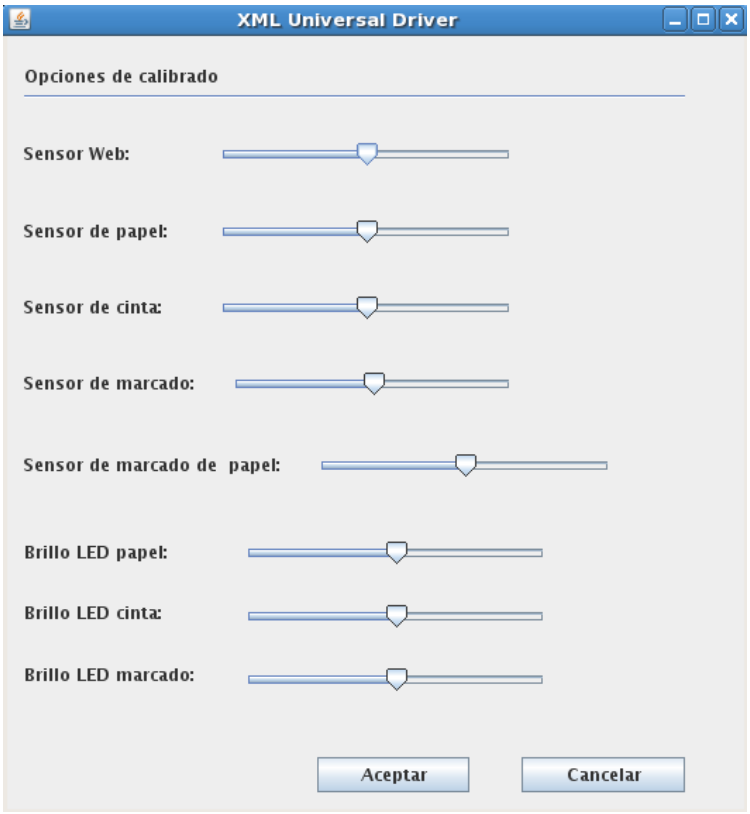

Figura D.42 Pantalla de opciones de calibrado para etiquetadoras

Una vez explicadas las opciones de configuración para las etiquetadoras, vamos a describir las opciones disponibles para las impresoras HP. La figura D.43 muestra estas opciones, que pueden ser consultadas en los manuales proporcionados por el fabricante para obtener más detalles. Hay opciones generales, como la selección de las bandejas de entrada y salida o activar la alimentación de hojas para que se haga de forma automática. Las opciones de ahorro son el grado de intensidad aplicable al texto y la densidad del tóner. Su valor comprende valores de 1 a 5, siendo 5 el máximo y el que más consumo implica. Las opciones de color se pueden modificar para variar la intensidad del color en general y también de cada color por separado si se pulsa el botón "Ver colores". Estos colores se muestran en la pantalla de la figura D.44. Por último se muestra la opción de alinear los cartuchos, tanto horizontalmente como verticalmente. Al igual que en las opciones de configuración de etiquetadoras se ofrece la posibilidad de guardar las opciones seleccionadas o volver a los valores de fábrica. En cualquier caso es posible imprimir una página de prueba tantas veces como se desee, pulsando el botón "Imprimir página de prueba", para comprobar que la configuración seleccionada es correcta. Si la nueva configuración es aceptable se pulsará el botón "Aceptar" para guardarla. Si se pulsa el botón "Cancelar" se conservará la configuración que existía previamente.

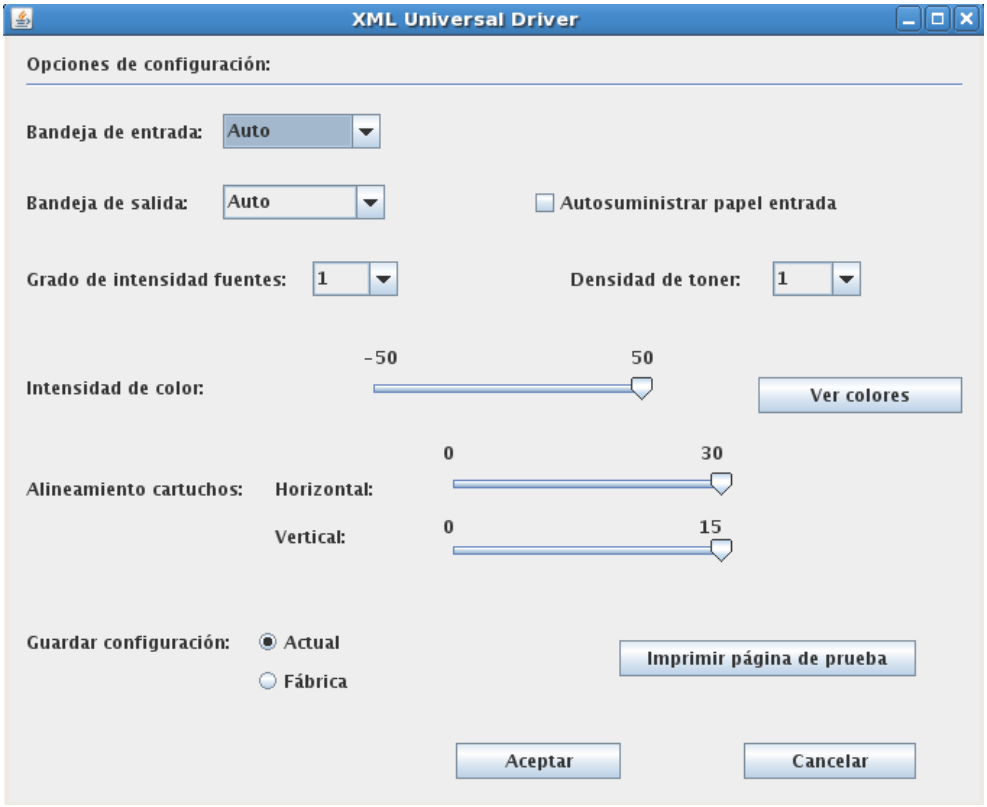

Figura D.43 Pantalla de opciones de impresión para impresoras HP

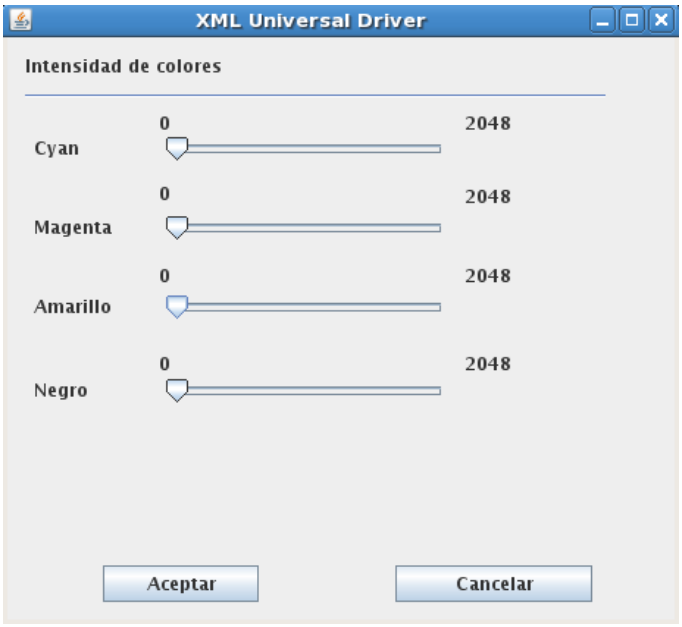

Figura D.44 Pantalla de opciones de colores para impresoras HP

Anexo E

Manual de instalación

En este anexo se explica el proceso de instalación del sistema. Se divide en dos secciones. En la sección E.1 se describe la instalación de la aplicación front-end. En la sección E.2 se explica la instalación de la aplicación motor.

E.1 Instalación de la aplicación *front-end*

En esta sección se describen los requisitos y pasos necesarios para la instalación de la aplicación *front-end*.

E.1.1 Requisitos para la instalación

Para instalar la aplicación *front-end*, se requiere:

- Una conexión a internet para el envío de mensajes con la aplicación *motor*.
- La máquina virtual Java, con una versión superior a 1.4.2, y su kit de herramientas *JDK*, con una versión superior a 6.25.

En este caso se ha utilizado la máquina virtual java con versión 1.4.2 y el kit *JDK* correspondiente a la versión 6.26. Si no se dispone de este kit de herramientas para Java ni de la máquina virtual (JRE) se deberán descargar de la página oficial de Java: *http://java.com/es/download/*. La última actualización a fecha de entrega de este proyecto es la 6.31.

E.1.2 Instalación de Java

Lo normal es que se tenga instalado Java, ya que se usa con muchos navegadores y aplicaciones. En el caso de que no se disponga de la aplicación Java, a continuación se describe cómo hacerlo siguiendo unas instrucciones muy sencillas. El manual de instalación de Java se puede consultar también en su página oficial.

Se recomienda, antes de proceder con la instalación en línea, desactivar el cortafuegos de Internet. En algunos casos, la configuración del cortafuegos predeterminado se establece para rechazar todas las instalaciones automáticas o en línea, como la instalación en línea de Java. Si el cortafuegos no se configura correctamente, podría

impedir la operación de descarga/instalación de Java en determinadas circunstancias. Consulte las instrucciones del manual específico del cortafuegos de Internet para desactivarlo.

1. Se comienza descargando la aplicación Java en *Java.com/es.*
2. En la página de descarga de Firefox (o de otro navegador), haga clic en el botón de descarga para iniciar el proceso de instalación.
3. Aparecerá el cuadro de diálogo Descarga de archivos y le pedirá que guarde el archivo descargado.
4. El administrador de descargas de *Firefox* aparecerá para informarle de que la descarga ha finalizado.
5. Para ejecutar el instalador, haga clic en el archivo en la ventana del administrador de descargas o en la ubicación en la que normalmente guarde los archivos de descarga. Dependiendo de la configuración de seguridad, es posible que aparezcan cuadros de diálogo en los que se pida permiso para continuar. Confirme que desea continuar con la instalación.
6. Se iniciará el proceso de instalación. El instalador presenta una opción para ver el contrato de licencia. Haga clic en el botón *Instalar* para aceptar los términos de la licencia y continuar con la instalación.
7. Oracle colabora con empresas que ofrecen distintos productos. Es posible que el instalador le ofrezca la opción de instalar estos programas como parte de la instalación de Java. Una vez seleccionados los programas que desee, haga clic en el botón *Siguiente* para proseguir con el proceso de instalación.
8. Se abrirán varios cuadros de diálogo con información para completar las últimas etapas del proceso de instalación; haga clic en *Cerrar* en el último cuadro de diálogo.

E.1.3 Instalación de la aplicación

La instalación de la aplicación puede realizarse en cualquier sistema operativo, ya que es multiplataforma. A continuación se detalla su instalación en un sistema operativo Windows. Esta aplicación no debería de instalarse aparte del producto SILO, ya que ha sido diseñada para formar parte de él. Aquí se explica cómo sería su instalación si no se incluyera en el paquete de instalación de SILO.

1. Se debe disponer de un *cd* que contenga la aplicación.
2. Una vez introducido el cd en el lector de su ordenador, debe acceder a la carpeta “XMLdriver”, que es la que contiene todos los archivos y el ejecutable de la aplicación.
3. Al abrir la carpeta “XMLdriver” aparecen varias carpetas y un archivo ejecutable java *.jar*.
4. Pulsando con un doble click sobre el ejecutable *driver.jar* se iniciará la aplicación..

El modo normal de instalación de esta aplicación será junto con la aplicación SILO, pero aquí no vamos a entrar en más detalles sobre la instalación de SILO.

E.2 Instalación de la aplicación motor

En esta sección se describen los requisitos y pasos necesarios para la instalación de la aplicación *motor*.

E.1.1 Requisitos para la instalación

Para instalar la aplicación *motor*, se requiere:

- Una conexión a internet para el envío de mensajes con la aplicación *front-end* y el resto de módulos del producto SILO.
- El servidor de base de datos de la aplicación SILO, que es un servidor Oracle correspondiente a la versión 10g Enterprise Edition.
- El sistema operativo Red Hat Linux Enterprise con una versión superior o igual a 5.0. El sistema operativo es un requisito que viene dado por la aplicación SILO, ya que esta aplicación está diseñada para formar parte del producto SILO.
- Los paquetes de librerías libxml2 y libxslt.

De estos requisitos se deduce que esta aplicación debe instalarse integrada junto con la aplicación SILO o en un entorno en el que SILO ya esté instalada.

E.1.2 Instalación de librerías

En esta sección se explica cómo instalar las librerías necesarias para que la aplicación pueda funcionar correctamente.

Primero explicaremos qué es *libxml2* y a continuación cómo instalarla. *Libxml2* es un conjunto de herramientas desarrollado para la plataforma *Gnome*, pero que puede ser utilizado en cualquier plataforma de forma libre. Es un *parser* XML implementado en el lenguaje C. Esta librería se considera muy portable y funciona sin problemas es una gran variedad de sistemas operativos.

El primer paso para instalar *libxml2* es acceder a su página oficial: *http://xmlsoft.org/*. Una vez que estemos en la página, hacer click en *Downloads* (Descargas). En esta sección de descargas pulsar sobre el enlace *xmlsoft.org*, que nos conducirá a sus servidor FTP: *ftp://xmlsoft.org/libxml2/*. En este servidor están disponibles todas las versiones desarrolladas desde 2004, incluida la librería *libxml*. Descargamos la última versión disponible de *libxml2*, que es la versión 2.7.8. Se pulsa sobre el paquete *libxml2-2.7.8.tar.gz* y se descarga. Una vez descargado, se necesita también su paquete

RPM correspondiente, que descargaremos pulsando sobre *libxml-devel-2.7.8-x86_64.rpm*. Ahora vamos a explicar cómo instalar estos paquetes en Linux. Primero se acede al directorio donde se han descargado los paquetes y a continuación ejecutamos una serie de comandos para su instalación.

El símbolo '#' representa al ***prompt***, no forma parte de los comandos. El proceso para compilar *libxml2* sería el siguiente:

1. Descomprimir los archivos:

   ```
   # tar -xvf libxml2-2.7.8.tar.gz

   # cd libxml-2-2.7.8
   ```

2. Para ver las opciones de compilación ejecutar:

   ```
   # ./configure --help
   ```

3. A continuación pueden verse todas las posibilidades que nos ofrece la configuración, ejecutando la ayuda:

   ```
   # ./configure [possible options]

   # make

   # make install

   # ldconfig
   ```

4. Se instala el paquete RPM de libxml2:

   ```
   # rpm -Uvh libxml-devel-2.7.8-x86_64.rpm
   ```

Completados todos estos pasos, ya tenemos lista la librería *libxml2*.

Los primeros pasos para instalar *libxslt* son los mismos que hemos seguido para la instalación de *libxml2*. Realizamos las mismas acciones hasta acceder al servidor FTP de *Gnome*: *ftp://xmlsoft.org/libxml2/*. Se puede ver que también están disponibles los paquetes correspondientes a la librería *libxslt*. Al igual que en el caso anterior, se descarga la última versión disponible pulsando sobre el enlace *libxslt-1.1.26.tar.gz* y a continuación sobre su paquete RPM: *libxslt-devel-1.1.26-1.x86_64.rpm.* Ahora vamos a explicar cómo instalar estos paquetes en Linux. Primero se acede al directorio donde se han descargado los paquetes y a continuación ejecutamos una serie de comandos para su instalación.

El símbolo '#' representa al ***prompt***, no forma parte de los comandos. El proceso para compilar *libxml2* sería el siguiente:

1. Descomprimir los archivos:

    # tar -xvf *libxslt-1.1.26*.tar.gz

    # cd *libxslt-1.1.26*

2. Para ver las opciones de compilación ejecutar:

    # ./configure --help

3. A continuación pueden verse todas las posibilidades que nos ofrece la configuración, ejecutando la ayuda:

    # ./configure [possible options]

    # make

    # make install

    # ldconfig

4. Se instala el paquete RPM de libxml2:

    # rpm -Uvh libxslt-devel-1.1.26-1.x86_64.rpm

Completados todos estos pasos, ya tenemos lista la librería *libxslt*.

E.1.3 Instalación de la aplicación

En esta sección se va a explicar su instalación en un entorno Linux donde ya se encuentran instalados todos los componentes del producto SILO. En el caso de no disponer de la aplicación SILO, no podrá funcionar correctamente, ya que lo mínimo que necesita es acceder al servidor de la base de datos de SILO. Como es HP la que se encarga del mantenimiento de su producto SILO, será la que instale esta nueva aplicación.

Partimos de que tenemos instalado el producto SILO. La instalación de la aplicación *motor* añadirá nuevas funcionalidades básicas a SILO. Los pasos a seguir son muy sencillos. En primer lugar, se descarga la carpeta "XMLdriver", que contiene todos los archivos necesarios para su instalación. Se guarda en uno de los directorios disponibles. Se abre un terminal desde Linux y se busca el directorio que contiene la carpeta "XMLdriver". A continuación se deben ejecutar los siguientes comandos en el terminal:

1. Se accede a los archivos de la carpeta:

    # cd XMLdriver

2. Se compilan los archivos para obtener el ejecutable de la aplicación:

```
# make –f proc.mk motorImp
```

3. Se ejecuta la aplicación:

```
# ./motorImp
```

A partir de ese momento el servidor de la aplicación permanecerá a la escucha y atenderá las peticiones que reciba.

Bibliografía

[1] David Hunter, Jeff Rafter, Joe Fawcett, Eric van der Vlist, Danny Ayers, Jon Duckett, Andrew Watt, Linda McKinnon. Beginning XML. John Wiley, 4ª edición.

[2] Ian Williams. Beginning XSLT and XPath: Transforming XML Documents and Data. John Wiley, 2009.

[3] Doug Tidwell. XSLT. O'Reilly Media, Inc., 2008.

[4] World Wide Web Consortium (W3C). Tutorial XML. http://www.w3.org/XML/

[5] W3C. Tutorial XML Schema. http://www.w3.org/XML/Schema

[6] W3C. Tutorial XSLT. http://www.w3.org/TR/xslt

[7] Tutorial XSL. http://www.w3schools.com/xsl/xsl_languages.asp

[8] Tutorial libxml2. http://xmlsoft.org/

[9] Tutorial libxslt. http://xmlsoft.org/XSLT/tutorial/libxslttutorial.html

[10] Tutorial Java. http://docs.oracle.com/javase/tutorial/index.html

[11] Oracle. PL/SQL. http://www.oracle.com/technetwork/database/features/plsql/index.html

[12] Sockets en C. http://www.eslinux.com/articulos/8591/programacion-sockets-lenguaje-c

[13] Sockets en Java. http://docs.oracle.com/javase/tutorial/networking/sockets/

[14] Lenguaje de impresión PCL. http://pcl.to/reference/

[15] Guía programación PCL / PJL. http://h20000.www2.hp.com/bc/docs/support

[16] Guía programación ZPLII. http://www.zebra.com/id/zebra/na/en/index/resource_library/manuals.html

[17] Guía programación PGL. http://www.printronix.com/products/manuals.aspx

[18] Wikipedia. Ejemplo del sistema de impresión CUPS (*Common Unix Printing System*). http://es.wikipedia.org/wiki/Common_Unix_Printing_System

[19] Wikipedia. Proceso Unificado. http://es.wikipedia.org/wiki/Proceso_Unificado