Instituto de Computación – Facultad de Ingeniería
Universidad de la República
Montevideo, Uruguay

PROYECTO DE GRADO INGENIERÍA EN COMPUTACIÓN

EDITOR DE HOJAS DE CÁLCULO WEB PARA SUITE OFIMÁTICA OPENGOO

Noviembre 2010

JUAN PEDRO DEL CAMPO

IGNACIO VÁZQUEZ

FERNANDO RODRÍGUEZ

Supervisor: Eduardo Fernández
Co-Supervisor: Marcos Saiz

RESUMEN DEL PROYECTO

El proyecto consiste en el desarrollo de una herramienta Web para la edición de hojas de cálculo denominada GelSheet, operada a través de un navegador de Internet y elaborada bajo las normas de código abierto (licenciamiento GPL II (1)).

El proyecto surge como iniciativa de los clientes de este proyecto, creadores de la aplicación ofimática *OpenGoo* (actualmente *Feng Office*), para ser integrada como un módulo adicional y ser extendida por parte de la comunidad de software libre. OpenGoo surgió como proyecto de grado de la Facultad de Ingeniería de la Universidad de la República (UDELAR), y ha logrado un importante impacto en la comunidad *Open Source*, posicionándose con el correr de los años como una de las herramientas líderes en el área.

GelSheet incorpora muchas de las características de las aplicaciones de escritorio convencionales aprovechando la aparición de las tecnologías Web 2.0, que facilitan la elaboración de aplicaciones interactivas. Para su desarrollo se enfatizó tanto en la interfaz gráfica, como en aspectos de rendimiento.

El resultado es una aplicación que implementa las funcionalidades básicas de forma eficiente, con una interfaz de usuario rica, que si bien debe ser extendida, proporciona una plataforma completa para su evolución.

ÍNDICE

1 INTRODUCCIÓN

Este capítulo presenta un resumen del proyecto, definiendo las características de la aplicación a realizar y las etapas llevadas a cabo durante su desarrollo.

1.1 MOTIVACIÓN

La aparición del concepto de Web 2.0 (2), dio comienzo a una iniciativa muy importante para el desarrollo de aplicaciones Web que posibilita explotar una gama de funcionalidades, que sólo se conseguían con aplicaciones de escritorio. La Web 2.0 utiliza tecnologías que permiten mediante técnicas de desarrollo, generar aplicaciones interactivas. Éstas se ejecutan del lado del cliente, es decir, en el navegador del usuario y mantienen una comunicación asíncrona con el servidor, redundando en interfaces de usuario más amigables, con una mejor interacción, más veloces y eficientes.

Es claro ver las ventajas que ofrecen las aplicaciones Web, como ser:

- Concurrencia.
- Movilidad que ofrece al usuario.
- No requiere instalación en el cliente.
- Centralización de la información.
- Desacoplamiento de sistemas operativos (multi-plataforma).

Se suma además el fácil mantenimiento de las aplicaciones, con beneficios tales como la propagación automática e instantánea de las actualizaciones.

Dadas estas ventajas proporcionadas por la Web, se comienza a apreciar una tendencia a migrar aplicaciones de escritorio hacia aplicaciones Web, que antes no eran posibles de implementar por limitaciones tecnológicas. Una rama de aplicaciones que ha cobrado especial interés, han sido las *suite*s ofimáticas, las que concentran herramientas para la organización, creación y modificación de archivos y documentos, en particular las planillas de cálculo.

OpenGoo (3) nace como un producto que busca proporcionar una solución Web ofimática de código abierto. Esta herramienta contempla buena porción de los componentes ofimáticos frecuentes, pero no cuenta con un editor de hojas de cálculo. Una cualidad destacable que posee *OpenGoo*, es que cuenta con el apoyo de una comunidad grande de usuarios y colaboradores, dispuestos a contribuir con el desarrollo y mejoras del producto.

El manejo de hojas de cálculo es uno de los pilares fundamentales en un producto ofimático, y es una de las funcionalidades más solicitadas por parte de los usuarios de dichas aplicaciones. Si bien existe un auge en la aparición de aplicaciones Web, la calidad de las aplicaciones de código abierto de hojas de cálculo, no cumplían, al momento del relevamiento, con las expectativas ni satisfacción de los usuarios.

Una limitante en el desarrollo de aplicaciones ofimáticas radicaba en la complejidad presentada en la exportación/importación de formatos de archivo de diferentes productos de este tipo. Gran parte de los formatos existentes eran “cerrados”, conocidos únicamente por

los propietarios. En los últimos años nace un consenso mundial por estandarizar y liberar los formatos de documentos, como forma de asegurar la interoperabilidad y portabilidad entre los diferentes sistemas de software. Surgen entonces, como respuesta a este fenómeno, formatos abiertos propuestos por las empresas de software que dominan el mercado, facilitando el desarrollo y la integración de documentos por parte de terceros.

GelSheet surge como alternativa a varias de las hojas de cálculo en línea que actualmente se encuentran disponibles, brindando la posibilidad de editar hojas de cálculo, con las ventajas de ser de libre distribución y uso; atributos no existentes en aplicaciones de edición similares.

1.2 OBJETIVOS

El objetivo es desarrollar una aplicación Web para la edición de hojas de cálculo, haciendo hincapié en el desarrollo de una interfaz amigable e intuitiva, con un tiempo de respuesta aceptable, y que permita un conjunto amplio de funcionalidades básicas para ser utilizado de forma satisfactoria por sus usuarios.

Se pretende que el producto sea diseñado de forma tal que pueda ser ejecutado de forma *stand-alone*, así como ser integrable a otros sistemas como módulo adicional, en particular al producto *OpenGoo*. Además este producto contará con las siguientes características: código abierto, facilidad de instalación y mantenimiento, compatibilidad entre navegadores, y que pueda ser extendido por parte de la comunidad. Deberá tener la capacidad de soportar diversos formatos de documentos, poniendo especial énfasis en aquellos de naturaleza abierta y así como los formatos más utilizados[1]. De la misma forma, *GelSheet* intenta posicionarse como la primera aplicación que cumpla con dichas funcionalidades, ya que no existen editores de hojas de cálculo en línea con estas características a la fecha.

No se busca elaborar un producto que abarque todas las funcionalidades de las aplicaciones líderes en el área, sino que se apunta a la construcción de una plataforma base extensible y adaptable, para que se puedan incorporar nuevos elementos de forma eficiente por terceros.

1.3 RESULTADOS

Se construyó una aplicación que cumple con gran parte de los objetivos establecidos, siendo evaluado de forma satisfactoria por parte del cliente.

El aspecto más destacable de la aplicación es la rica interfaz de usuario lograda, tanto en el aspecto visual como en la usabilidad de la misma. Para concebir una aplicación madura es necesario extender las funcionalidades y complementos tales como las gráficas, fórmulas e imágenes, así como mejorar aspectos como la robustez.

El diseño arquitectónico construido busca el uso eficiente de los recursos, permitiendo el manejo de grandes volúmenes de datos, con tiempos de respuesta razonables. La arquitectura modular llevada a cabo permite el agregado y extensión de sus funcionalidades de forma sencilla.

[1]Basado en el análisis del mercado realizado. Ver anexo "*Distribución del Mercado*".

Se logró la compatibilidad con la mayoría de los navegadores más utilizados por los usuarios, exceptuando al navegador Internet Explorer, para el cual es necesario corregir algunas deficiencias en la interacción con el usuario.

El producto obtenido cumple gran parte de los requerimientos iniciales establecidos al comienzo del proyecto, constituyendo un buen punto de partida para el desarrollo a futuro de una plataforma completa y estable.

1.4 Resumen de los Capítulos

En el capítulo 2 se presenta el Estado del Arte de la aplicación, desarrollando la historia de las hojas de cálculo y el análisis de las aplicaciones relevadas. El capítulo 3 detalla los requerimientos del producto, destacando tanto los requisitos funcionales como no funcionales del mismo. Al final de dicho capítulo, se expone un refinamiento del alcance, producto de negociaciones con el cliente. Continuando con la organización del documento, el capítulo 4 presenta el diseño de la interfaz del usuario, enumerando las principales funcionalidades de la misma. Los aspectos técnicos del desarrollo están expuestos en los capítulos 5 y 6 dónde se presentan la arquitectura de la aplicación y las decisiones de diseño del producto respectivamente. El capítulo 7 abarca en forma esquemática los lineamientos básicos de la implementación del producto desarrollado, mientras que el capítulo 8 expone las conclusiones y una propuesta del trabajo futuro a realizar.

2 ESTADO DEL ARTE

Este capítulo se compone de dos secciones. En la primera sección se hará un repaso de la evolución de las aplicaciones de hojas de cálculo, desde sus comienzos hasta el día de hoy, finalizando con el surgimiento de las *spreadsheets online*. En la segunda sección se presenta un resumen de las diferentes aplicaciones para la edición de hojas de cálculo que fueron relevadas, tanto Web como de escritorio. Se evaluarán los puntos más destacados de cada una y se contrastarán sus virtudes con sus defectos.

2.1 HOJAS DE CÁLCULO

En la terminología contable, una hoja de cálculo o *spreadsheet*, es simplemente una tabla (o matriz) de filas y columnas, utilizada para el manejo de información. La intersección entre cada fila y columna determina una celda. Una celda puede contener cualquier tipo de datos, incluyendo: números, fórmulas y textos.

Producto de los avances de la informática surgió la iniciativa de desarrollo de planillas electrónicas, cuya diferencia frente a las convencionales es su flexibilidad, velocidad y precisión que proveen.

A la matriz de celdas suele denominársele "hoja", y por lo general un documento está compuesto por varias de éstas, el cual se denomina "libro".

Dentro de la hoja, las filas suelen etiquetarse con números, mientras que las columnas con caracteres alfabéticos. Las celdas quedan identificadas por la dupla (columna, fila). A su vez las hojas son identificadas mediante un nombre arbitrario al igual que el libro. Estas etiquetas suelen denominarse "direcciones".

Para referenciar a una celda se define una "dirección", compuesta por la jerarquía Libro->Hoja->Columna->Fila. La sintaxis tradicional para una dirección genérica sería [Libro]Hoja!ColumnaFila, aunque suele ser omitido el componente libro para referencias dentro del mismo documento, y también se omite el nombre de la hoja para referencias dentro de la misma.

Un rango es un bloque rectangular de una o más celdas, que la hoja de cálculo trata como una unidad. De manera convencional el rango queda determinado mediante dos direcciones: donde comienza y donde termina, separadas por ":". Por ejemplo si se desea referirse a la columna "B", el rango se denota "B:B", para la tercera fila "3:3", y para un conjunto de celdas finito "B6:J8" (conjunto de celdas determinados por las coordenadas de sus vértices superior izquierdo e inferior derecho).

Las operaciones en una hoja de cálculo pueden ser aplicadas ya sea a referencias directas a celdas como a conjuntos de celdas (rangos), por lo cual estos adquieren vital importancia dentro de las hojas de cálculo. De hecho, de no existir los mismos, operaciones aplicadas a columnas, filas, o un conjunto grande de celdas serían demasiado complejas de definir. Por ejemplo la suma de una columna, que de otra manera debiese escribirse como la suma de cada una de sus celdas individuales.

Por lo general las aplicaciones proveen formas de referenciar a rangos de manera más amigables mediante la utilización de "nombres" definidos por el usuario. Éstos no son otra cosa que un nombre que se le asigna a una dirección, por ejemplo, es posible dar el nombre "Facturas" al rango "A1:F19" de manera de referenciar a una tabla, "IVA" al nombre de una celda de forma de hacer más legibles las fórmulas que las referencien ("=100*IVA").

Como se mencionó anteriormente las celdas pueden contener datos simples como valores numéricos y alfanuméricos, pero también pueden contener "*fórmulas*". Las fórmulas son expresiones matemáticas compuestas de operadores y un conjunto de funciones (eventualmente una), que operan sobre los tipos de datos básicos y/o sobre rangos. Para diferenciar un valor de una fórmula, generalmente se suele comenzar con un carácter especial, típicamente "=", y/o"+" o "-".

El resultado es desplegado en la misma celda, con lo cual las aplicaciones suelen tener definidos 2 atributos básicos por celda, la fórmula y el valor retornado.

Es frecuente que las aplicaciones incorporen componentes adicionales, como son gráficas (que extraen datos de rangos), imágenes y otros.

2.2 HISTORIA DE LAS HOJAS DE CÁLCULO

Los comienzos de las *spreadsheets* u hojas de cálculo electrónicas se remontan al año 1978 cuando Daniel Bricklin en conjunto con Bob Frankston desarrollan la idea de implementar una calculadora interactiva. Esta idea se plasma con la creación de VisiCalc, una aplicación para computadoras personales que se considera la pionera en la edición de *spreadsheets*. El prototipo de VisiCalc elaborado por Frankston era capaz de realizar operaciones en una matriz o grilla de 20 filas y 5 columnas.

VisiCalc (acrónimo de *Visible Calculator*) es comercializada por Bricklin y Frankston a través de Software Arts Corporation.

Al comienzo de la década de los '80, Mitch Kapor, desarrolla Lotus 1-2-3, la primera aplicación para la edición de *spreadsheets* que incorpora varios de los conceptos que nos llegan hasta el día de hoy. Entre éstos se destacan: el nombrado de celdas, manejo de rangos y utilización de macros. Además, incorpora nuevas funcionalidades, tales como el uso de gráficas, y soporte a bases de datos.

Lotus Development, la compañía de Mitch Kapor, adquiere Software Arts y VisiCalc es discontinuado.

En 1984 surge la primera versión de Microsoft Excel, desarrollada para Machintosh, el modelo de computadora personal desarrollado por Apple. Fue una de las primeras aplicaciones de este tipo en utilizar interfaz gráfica con menús emergentes e interacción con el mouse. En 1987, con el lanzamiento de Microsoft Windows, Excel se convirtió en una de las aplicaciones estrella de este sistema operativo. Excel fue el único editor de *spreadsheets* de esta plataforma hasta 1992, y hasta el día de hoy continua siendo la aplicación líder de este rubro en el mercado. Surgirían otras aplicaciones que alcanzaron un éxito relativo en el mercado entre 1985 y los comienzos de los '90. Entre ellas la que más se destaca es Quattro de la empresa Borland, que competiría por el mercado de las *spreadsheets* en las

computadoras personales, junto a Lotus 1-2-3 y Microsoft Excel. Otra de las aplicaciones que surge en ese momento es StarCalc componente de StarOffice, desarrollado por la empresa StarDivision, con un formato de interacción similar al de Microsoft Excel.

StarDivision es adquirida por Sun Microsystems en 1999, con el objetivo de competir con Microsoft Office, la *suite* de aplicaciones de escritorio de Microsoft. Más adelante el código de StarOffice sería liberado por Sun, creando la *suite* OpenOffice. Esta *suite* está integrada por un conjunto de aplicaciones, entre ellas OpenOffice Calc, la versión libre de StarCalc, pudiendo la misma ejecutarse tanto en ambientes Windows como Linux entre otros (4).

Con la mejora de las tecnologías aplicadas a Internet desde la década del '90, comienza la aparición de herramientas y lenguajes para la carga de contenidos y comunicación asíncrona en los navegadores de Internet. En 2006 la Word Wide Web Consortium elabora la primera especificación de lo que hoy en día se conoce como AJAX, una combinación de tecnologías existentes para la creación de sitios Web interactivos. Básicamente AJAX (*Asynchronous Javascript And XML*) es una tecnología que permite al navegador realizar peticiones HTTP sin la necesidad de recargar toda la página (5). Dichos avances permitieron la creación de diversas aplicaciones para la creación y edición de *spreadsheets* a través de un navegador de Internet.

En estos últimos años, varias empresas han orientado sus desarrollos Web en la creación de *Web Suit*es, estando dentro de las mismas, las *spreadsheets*. Al día de hoy existen diversas aplicaciones Web que comienzan a competir en varios aspectos con las aplicaciones de escritorio (6), destacándose "Google *Spreadsheet*" de Google Inc., "Zoho Sheet" de Zoho Corp., "EditGrid" de EditGrid Team. Ninguna de estas de código abierto.

2.3 APLICACIONES RELEVADAS

2.3.1 INTRODUCCIÓN

En esta sección se resume las características de las aplicaciones de hojas de cálculo que se encuentran actualmente en el mercado, tanto Web como de escritorio. De esta manera se hará una comparación entre los diferentes productos, y se determinarán los puntos que la aplicación a desarrollar debe soportar, luego de una evaluación y clasificación de los mismos.

2.3.2 RESUMEN DE LAS APLICACIONES RELEVADAS

De las aplicaciones evaluadas, se eligió las cinco aplicaciones líderes en el mercado (7), tres Web y dos de escritorio.

- Aplicaciones de Escritorio:
    - Microsoft Excel 2003
    - OpenOffice Calc 2.4.0 (*release* 27/03/2008)
- Aplicaciones Web:
    - Google Spreadsheets (*release* 23/06/2008)
    - Zoho Sheet (*release* 13/06/2008)
    - EditGrid (*release* 19/06/2008)

Actualmente las aplicaciones de escritorio prevalecen ante las Web, ya que poseen mayores funcionalidades y a su vez un mejor rendimiento, lo que deriva en mejores tiempos de respuesta. En particular Microsoft Excel es la que lidera el mercado general, mientras que Google Spreadsheet lo hace en el mercado Web.

Para ahondar en un análisis detallado de cada una de las aplicaciones relevadas se recomienda leer el "*Anexo – Aplicaciones Relevadas*"

Como puntos a destacar, las aplicaciones de escritorio soportan una mayor cantidad de datos (volumen), operan con mayor eficiencia y poseen mayores funcionalidades. Por otro lado las aplicaciones Web poseen características colaborativas que las de escritorio no, además de proporcionar mejor accesibilidad a los documentos (disponibilidad global).

Durante la etapa de relevamiento, se estudió la posibilidad de incluir a Microsoft Excel 2007 en la lista de aplicaciones a evaluar. Finalmente, se decidió no relevar este producto por las siguientes razones:

- Excel 2007 introduce un nuevo paradigma en el diseño del menú de opciones para el usuario, denominado "cinta de opciones". Esta "cinta de opciones", modifica la presentación y despliegue del menú, al igual que el agrupamiento de las opciones de edición disponibles (8). Esta nueva modalidad, no se ajusta al perfil de interfaz buscado, ya que se deseaba mantener la homogeneidad de las interfaces de usuario que sí poseen las aplicaciones seleccionadas.
- Esta versión no añade grandes cambios con respecto a las funcionalidades de la versión 2003 que sí fue escogida.
- Al momento de realizar el estudio, Excel 2007 presentaba una escasa participación en el mercado en relación a la versión 2003 relevada.

En la Tabla 1, se presenta un comparativo de las diferentes aplicaciones relevadas durante la etapa de análisis del estado del arte.

	De escritorio		Web		
	MS Excel 2003	**OpenOffice Calc**	**Google *Spreadsheets***	**Zoho Sheet**	**EditGrid**
Volumen	• 65536 filas x 256 columnas / hoja • Sin límite de hojas / libro	• 65536 filas x 256 columnas / hoja • Hasta 256 hojas/libro	• 256 Columnas200000 Celdas • Sin límite de filas	• Sin limite	• 65536 filas x 256 columnas / hoja • Cantidad ilimitada de hojas / libro • Permite importar hasta 8Mb de datos en la versión paga y 2Mb gratuitos
Seguridad	• Protección a nivel de libro, celdas y rangos	• Protección a nivel de libro, celdas y rangos	• Protección a nivel de libro	• Protección a nivel de libro, celdas y formulas	• Protección a nivel de libro, celdas y rangos
Colaboración	• Si (Sin cambios en tiempo real, limitada)	• Si (Sin cambios en tiempo real, limitada)	• Concurrencia en tiempo real	• Concepto de grupos • Concurrencia en tiempo real	• Concepto de grupos • Concurrencia en tiempo real
Operaciones	• 15 dígitos de precisión • Anidamiento entre funciones	• Anidamiento entre funciones	• Categorización de funciones similar a Excel • Funciones propias de google	• Categorización similar a Excel y google *spreadsheets*	• Soporta más de 500 funciones
Formatos	• 40 formatos distintos • No compatible con ODF • No exporta a PDF	• 19 formatos distintos • Utiliza formatos estándares para el almacenamiento por defecto • Exportación a PDF en forma nativa • Soporta OOXML	• 5 formatos distintos (propietarios y no propietarios) • Compatible con PDF	• Exportación a: XLS, ODS, CSV, HTML, GNUmeric y PDF • Importa XLS, CSV, ODS	• Exportación a: XLS, HTML, CSV, PDF, ODS, SXC, GNUmeric, TeX, XML • Importación: XLS , ODS, GNUmeric, Lotus123, SXC, CSV
Usabilidad	• Intuitiva • Diversidad de teclas de acceso rápido	• Semejante a MS-Excel • Teclas de acceso rápido • Complejo uso de corrector ortográfico	• Aplicación sencilla e intuitiva para el usuario promedio • Soporte a teclas de acceso directo limitadas • Fuentes limitadas	• Similar a MS-Excel • No cuenta con barra de menú • 20 Tipos de fuentes. • Teclas de acceso rápido	• Interacción fluida • Interfaz sencilla, intuitiva • Teclas de acceso rápido
Imágenes	• 20 formatos distintos • Soporte de autoformas e integración con componentes MS	• 14 formatos distintos • Soporte de elementos OLE	• Vía URL • Editor de imágenes integrado	• Soporte de formatos JPG y PNG desplegadas en paneles flotantes • Permite insertar imágenes vía url	• Vía URL
Gráficas	• 14 tipos de graficas estándar • 56448 combinaciones de diseños.	• 9 tipos 2D y 3D	• 6 tipos distintos	• 9 tipos distintos • Publicación de graficas en la Web	• 12 Categorías diferentes
Licencia	• Propietario	• LGPL	• Propietario	• Propietario	• Propietario
Sistema Operativo	• Windows	• Multiplataforma	• En línea	• En línea	• En línea
Costo	• US$ 111 la versión estándar • US$ 134 la versión profesional	• Gratuito	• Gratuito	• Gratuito para uso personal • Variable para uso empresarial	• Gratuito

Tabla 1 - Planilla comparativa de las aplicaciones relevadas

2.3.3 *CONCLUSIONES DEL RELEVAMIENTO*

Durante la etapa de relevamiento de aplicaciones se constató que no existían productos de código abierto similares al propuesto en el proyecto de grado. Los editores de hojas de cálculo en línea relevados son de licencia propietaria, por lo que no se dispone de acceso al código fuente del mismo. Esto es de gran importancia al momento de modificar, mantener y extender la aplicación por parte de la comunidad. Por otro lado, las aplicaciones de código abierto encontradas no cumplen con las funcionalidades mínimas para su utilización, siendo muy primitivas y descartadas para utilizar como base de este proyecto.

La posibilidad de utilizar *GelSheet* como complemento de otra aplicación existente, también es una característica única de la aplicación, y no se encuentra como funcionalidad en ninguna de las aplicaciones relevadas. Esta característica, es un punto fuerte de la aplicación y está basada en el creciente fenómeno de desarrollo de *plugins* o complementos para la mejora y extensión de funcionalidades de aplicaciones de mayor porte (como por ejemplo el FCK Editor, editor de texto).

En base al relevamiento de las aplicaciones realizado, se determinaron las características que deben ser tomadas en cuenta para el desarrollo:

- Interfaz de Usuario atractiva y similar a las interfaces vistas.
- Uso de la aplicación intuitivo y amigable.
- Compatible con formatos de documento de aplicaciones de escritorio.
- Funcionalidades robustas.
- Rápido nivel de respuesta de la aplicación.
- Concurrencia entre usuarios.

Sin estos requisitos, no se estaría a la altura de las aplicaciones más utilizadas del mercado, por lo cual son necesarios para la construcción de una aplicación de esta naturaleza.

Por tanto, tomando de las aplicaciones existentes los requerimientos básicos para la construcción y buen funcionamiento, y sumando las características únicas que se pretenden introducir en el producto, consideramos que el desarrollo de este proyecto puede ser un punto de partida para la incorporación de editores de hojas de cálculo en las aplicaciones de código abierto.

2.4 NAVEGADORES

2.4.1 *PRINCIPALES NAVEGADORES*

Esta sección se exponen los principales navegadores, sus características, desempeño y compatibilidad con los estándares Web. La elección de los navegadores evaluados radica en función de la distribución del mercado, es decir de la cantidad de usuarios que lo utilizan. Para ahondar en mayor detalle sobre su posicionamiento en el mercado ver "*Anexo - Análisis del Mercado de Aplicaciones"*, sobre el análisis del rendimiento ver "*Anexo - Análisis de los Navegadores".*

Microsoft Internet Explorer:

Navegador de la empresa Microsoft incorporado en su producto Microsoft Windows. Al momento de realizar el relevamiento del estado del arte este navegador abarca la mayor parte del mercado, principalmente en sus versiones 6 y 7. Navegador compatible únicamente con la plataforma Microsoft Windows.

Si bien es el navegador más utilizado, su rendimiento principalmente en cuanto al procesamiento de Javascript, en comparación con sus pares es sensiblemente menor.

Otra desventaja que presenta este navegador para los desarrolladores es la compatibilidad con los estándares de la Web.

Mozilla Firefox

Producto de Mozilla Foundation de código abierto con creciente inserción en el mercado, siendo las principales versiones presentes en el momento del relevamiento la 2 y la 3. Compatible con los sistemas operativos Microsoft Windows, Mac OS/X y Linux entre otros.

Debido a su característica de código abierto existen varios productos derivados que utilizan su código con modificaciones particulares, como la optimización para su funcionamiento en diversas arquitecturas, diferentes interfaces, etc. Los productos más destacados son: IceWeasel, GNU IceCat, SwiftFox entre otros.

Google Chrome

Navegador de la empresa Google Inc. Con constante crecimiento en el mercado, es considerado el navegador más rápido del momento. Desarrollado para los ambientes Microsoft Windows, Mac OS/X y Linux, compilado con base en componentes de código abierto del proyecto Chromium.

Opera Browser

Navegador de internet de la empresa Opera Software, es considerado líder en el mercado para dispositivos móviles en sus versiones Opera Mobile y Opera Mini. Compatible con los sistemas Microsoft Windows, Mac OS/X y Linux entre otros.

Safari

Producto de Apple Inc. es un navegador Web disponible únicamente para los sistemas Mac OS/X y Microsoft Windows. Utiliza el motor de renderizado WebKit de código abierto creado por el proyecto KDE utilizado por el navegador Konqueror.

2.4.2 COMPATIBILIDAD

Es necesario resaltar la diferente interpretación de los lenguajes HTML, Javascript y CSS que realizan los navegadores, lo cual representa una gran dificultad al momento de implementar la aplicación. Debido a estas restricciones, es necesario aplicar soluciones no convencionales

y rebuscadas, que exigen la introducción de código adicional en los códigos fuente y las hojas de estilo. Dichas técnicas permiten alterar el comportamiento de la aplicación según el navegador en que se encuentre, añadiendo a su vez, un tiempo de prueba (*testing*) para cada navegador.

Contrariamente a las tendencias del mercado, el navegador más utilizado, Microsoft Internet Explorer, resulta el navegador más lento en términos de procesamiento Javascript, tiempos de acceso al DOM, rendimiento en la renderización. A su vez presenta grandes complejidades referentes a la compatibilidad, ya que no cumple totalmente con las principales recomendaciones propuestas por organizaciones tales como *World Wide Web Consortium* o *Ecma International*.

3 REQUERIMIENTOS

En este capítulo se describen los requerimientos que debe implementar el sistema a construir. El origen de los mismos surge del análisis de las aplicaciones relevadas en el Estado del Arte, en conjunto a los requisitos negociados con el cliente.

Los requerimientos son divididos en dos partes: aquellos referentes a las funcionalidades que debe implementar la aplicación (los llamados "Requerimientos Funcionales"), y por otro lado a las cualidades que deben ser parte del sistema, pero que no constituyen una funcionalidad en sí misma, como son rendimiento, plataformas, tiempo de respuesta, tecnologías, etc. ("Requerimientos No Funcionales").

A su vez se dedica un apartado en cada tipo de requerimiento que señala el enfoque y prioridades que fueron específicamente derivados del cliente.

Se concluye con un "Refinamiento del Alcance", en el cual se enumeran aquellas características que fueron relegadas del alcance preliminar, acordando previamente con el cliente.

3.1 REQUERIMIENTOS FUNCIONALES

Una vez en la etapa del Estado del Arte, se comenzó a elaborar un listado de las funcionalidades en común de cada una de las aplicaciones que compusieron dicho relevamiento.

Se especificó el criterio de priorización de las funcionalidades en común, teniendo en cuenta los siguientes factores: las funcionalidades mínimas especificadas por el cliente, la similitud con las aplicaciones relevadas, y el tiempo estimado de desarrollo de dichas funcionalidades.

Los requerimientos funcionales obtenidos, son los siguientes:

- Cada documento puede tener más de una hoja de cálculo.
- Soporte a referencias dentro de una hoja de cálculo o entre ellas, dentro del mismo documento (excluido entre documentos).
- Soporte a rangos (nombrado, navegación entre ellos).
- Soporte a navegación mediante periféricos de teclado y mouse.
- La aplicación utilizará combinaciones de teclas de acceso rápido.
- La cantidad de colores de la aplicación debe ser la soportada por Microsoft Excel 2003 y OpenOffice Calc.
- Los tipos de fuente soportados deben ser las fuentes comunes a Microsoft Excel 2003 y OpenOffice Calc.
- Los tipos de valores mínimos que debe soportar la aplicación son
    - Numérico (con separador de miles)
    - Texto
    - Moneda y símbolos ($, U$S, etc.)
    - Fecha
    - Hora

    - Porcentaje
    - Hipervínculo
- Debe soportar ajuste de celdas, reducción y combinación entre las mismas.
- Debe soportar formatos de edición (copiar, pegar, cortar).
- Herramienta de búsqueda y reemplazo de texto.
- Soporte a funcionalidad de deshacer y rehacer (un*do, redo*).
- Soporte de borrado de formato.
- Exportación del documento a los siguientes formatos:
    - Excel 2003 (.xls)
    - *Portable Document Format* (.pdf)
    - *Open Document Spreadsheet* (.ods)
    - *Extensible Markup Language* (.xml)
    - *HyperText Markup Language* (.html)
    - Texto (.csv).
- Importación de *spreadsheet* desde los siguientes formatos:
    - Excel 2003 (.xls)
    - *Open Document Spreadsheet* (.ods)
    - *Extensible Markup Language* (.xml)
    - Texto (.csv)
- Protección de edición del documento:
    - Por documento
    - Por rangos
    - Por celdas
- Detección de errores:
    - Referencias circulares
    - División entre cero
    - Referencias muertas
- Operaciones matemáticas entre rangos:
    - Sumatoria
    - Promedio
    - Máximo
    - Mínimo
    - Cantidad
- Soporte a creación de graficas:
    - Columnas
    - Barras
    - Áreas
- Edición en forma concurrente de una hoja de cálculo por varios usuarios

3.1.1 REQUERIMIENTOS ESPECÍFICOS DEL CLIENTE

En las primeras reuniones con el cliente, se estableció que el sistema a construir debería ser un producto con un enfoque competitivo en el área; un mercado de *spreadsheets online* con aplicaciones tales como Google Spreadsheet, Zoho Sheet y EditGrid; todas ellas utilizadas por un sector importante de la comunidad Web.

Se tomó como aplicación modelo a Google Spreadsheets, la aplicación con mayor aceptación de las mencionadas anteriormente[2].

3.2 REQUERIMIENTOS NO FUNCIONALES

Los requerimientos no funcionales obtenidos durante la etapa del Estado del Arte surgen de la evaluación y la utilización de las diferentes aplicaciones relevadas.

Los requerimientos no funcionales relevados en esta etapa son los siguientes:

- **Usabilidad**. La aplicación debe poder ser utilizada de manera intuitiva y de forma similar a las aplicaciones convencionales.
- **Rendimiento**. La aplicación debe tener tiempos de respuesta razonables en las operaciones típicas.
- **Tolerancia a fallos**. La aplicación debe manipular posibles errores de manera que no comprometan la información del usuario o incapacite su funcionamiento.
- **Estética**. Debe ser visualmente agradable y con un diseño similar a las aplicaciones relevadas.

3.2.1 REQUERIMIENTOS ESPECÍFICOS DEL CLIENTE

Primeramente, la aplicación debe construirse para ser integrada a *OpenGoo*, la *Web Suite* ya elaborada por el cliente, para formar parte de las aplicaciones en ella constituidas.

Otro de los requerimientos establecidos por el cliente es el de utilizar las mismas tecnologías base con las que se construyó la *Web Suite OpenGoo*. *OpenGoo* está construido utilizando HTML y Javascript en el lado del navegador y PHP en el lado del servidor. La persistencia se realiza utilizando el motor de base de datos MySQL Se apunta a que ambas aplicaciones interactúen de forma integrada y no requieran nuevas tecnologías.

Uno de los aspectos más importantes a la hora de construir una aplicación Web en general, es que los tiempos de respuesta de la misma, sean lo más rápido posibles. Este requisito es uno de los más importantes para la construcción del sistema, ya que una aplicación con una gran cantidad de funcionalidades pero con tiempos de respuesta lentos, no resultaría de utilidad para el usuario final.

La aplicación debe poder ejecutarse en los navegadores más utilizados del mercado, por lo que debe ser capaz de ajustarse a las diferentes características y variaciones que cada uno de los mismos tiene. Los navegadores elegidos fueron Microsoft Internet Explorer 7.0+ y Mozilla Firefox 3.0+ ya que la participación en el mercado entre Microsoft Internet Explorer y Mozilla Firefox superaba el 90% del mercado[3].

El cliente priorizó la robustez antes que la complejidad, es decir que estaba dispuesto a relegar funcionalidades en pro de la robustez de la aplicación. Por lo tanto, el sistema

[2] Conclusión tomada a partir del análisis de distribución del mercado de las aplicaciones Web ofimáticas. Ver anexos "*Distribución del Mercado*" por más detalles.
[3]Los datos son tomados en Mayo-Julio del 2008, según las fuentes de www.hitslink.com y www.w3schools.com. Ver anexos *"Distribución del Mercado"* por más detalles.

desarrollado debe ser extensible, de manera que el agregado de nuevas funcionalidades a la misma no requiriera un rediseño o cualquier forma de trabajo adicional.

3.3 Refinamiento del Alcance

Durante el período de elaboración de la aplicación, se tuvieron que realizar diferentes cambios al alcance preliminar descrito en las secciones anteriores.

Se estableció inicialmente un alcance primario de la aplicación, considerando las funcionalidades principales que debe tener la misma. Este alcance se realizó a partir del alcance preliminar. Posteriormente se realizó un refinado de requerimientos secundarios tomando en cuenta dos pautas:

Se relegaron requerimientos del alcance inicial debido a que se consideró que el tiempo de desarrollo de los mismos limitaría el tiempo para consolidar funcionalidades ya desarrolladas, premisa fundamental del cliente (robustez frente a complejidad).

Como primer punto se bajó la prioridad al soporte de varias hojas por libro, dejándolo como un requerimiento secundario, que debiese ser abordado una vez alcanzado las funcionalidades básicas. Los requisitos elementales establecidos son:

- Formato de fuente, selección de color y alineación de la misma.
- Soporte a fórmulas matemáticas y de estadística básicas.
- Impresión y exportación de hojas de cálculo. Impresión refiere a exportación a PDF.
- Soporte de teclado para la navegación (teclas de acceso directo).
- Control de errores.
- Persistencia.
- Seguridad.

Con respecto a aquellos relacionados con la variedad de formatos, se hizo una selección de los considerados más relevantes, para poder dedicar más tiempo a la elaboración de otras funcionalidades. En cuanto a los formatos de celdas, en primera instancia se decidió soportar únicamente campos numéricos y de texto, considerando para trabajos futuros el soporte para los formatos restantes.

Uno de los aspectos relegados en una etapa temprana del desarrollo fue la importación de hojas de cálculo hacia la aplicación. La decisión radica en que al no poder soportar de forma completa los diferentes tipos de formatos y de funcionalidades que disponen las aplicaciones de escritorio, se perdería en gran parte la fidelidad del documento importado.

Otro aspecto considerado importante que fue descartado del alcance final y puesto como mejora en futuros desarrollos fue la implementación de gráficas a partir de datos del documento. Se realizó un estudio, aunque no en profundidad, de posibles herramientas ya existentes para la creación de gráficas. Si bien se analizaron algunas alternativas como CanvasGraph (9), JpGraph (10)o PlotKit (11), se decidió utilizar el tiempo activo de proyecto para la corrección de funcionalidades ya establecidas.

La posibilidad de inmovilizar paneles y la división de una *spreadsheet* en varios paneles fue relegada debido a su complejidad.

La concurrencia es posible para la lectura de cualquier cantidad de usuarios, pero es parcial ya que no retroalimenta a los mismos de las acciones de otros. Por otro lado la edición no realiza bloqueos, y al guardar puede sobrescribir cambios de otro usuario. La concurrencia de edición de la hoja de cálculo por varios usuarios se quitó del alcance primario, debido a que se utilizó más tiempo del estimado en rediseños de los requerimientos prioritarios discutidos con el cliente.

4 GUI (GRAPHIC USER INTERFACE)

El objetivo de este capítulo es presentar los aspectos gráficos de la aplicación, previo a introducir los conceptos técnicos, que describen cómo se ha implementado la solución (12). Se hará una presentación de la interfaz de usuario de la aplicación, también llamada GUI por sus siglas en inglés (*Graphic User Interface*).

La Figura 1 muestra la interfaz gráfica que es presentada al usuario de la aplicación, también denominada como el "Área de trabajo".

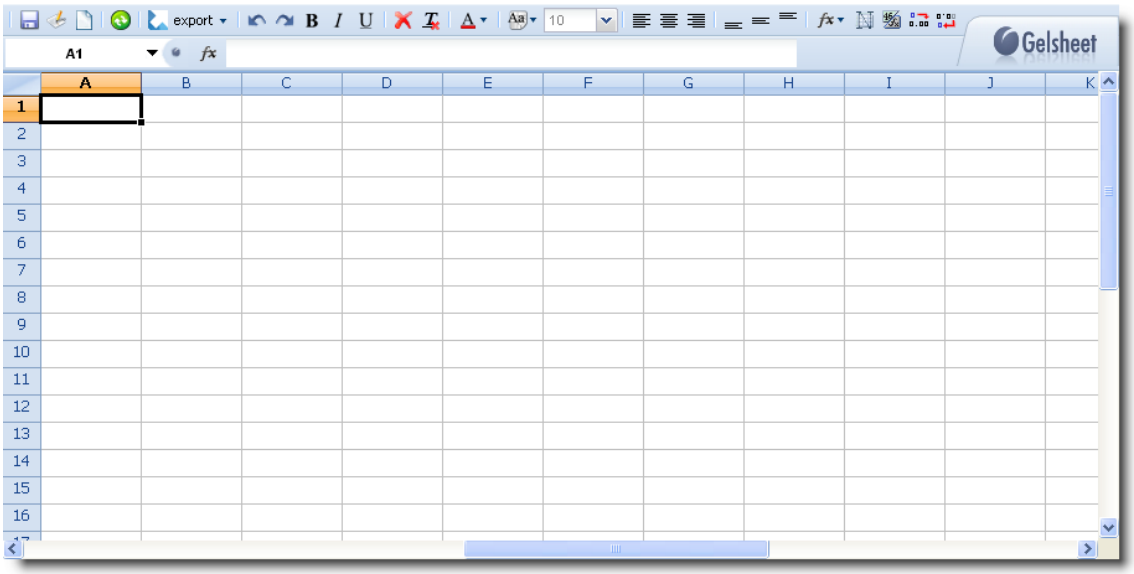

Figura 1 - Interfaz de Usuario de GelSheet

Para comprender mejor qué debe contemplar la interfaz de usuario de la aplicación, se expondrán los elementos manejados en el común de las aplicaciones de hojas de cálculo disponibles.

Los elementos usuales que el usuario dispone en este tipo de aplicaciones son:

- Barra de menús.
- Barras de Herramientas.
- Barra de fórmulas.
- Grilla (matriz de celdas).
- Otras Regiones:
  - Barra de estado.
  - Selector de Hojas del libro.
  - Regiones Auxiliares.

La Figura 2, permite explorar los componentes que son utilizados por Microsoft Excel 2003, que son esencialmente los descritos anteriormente.

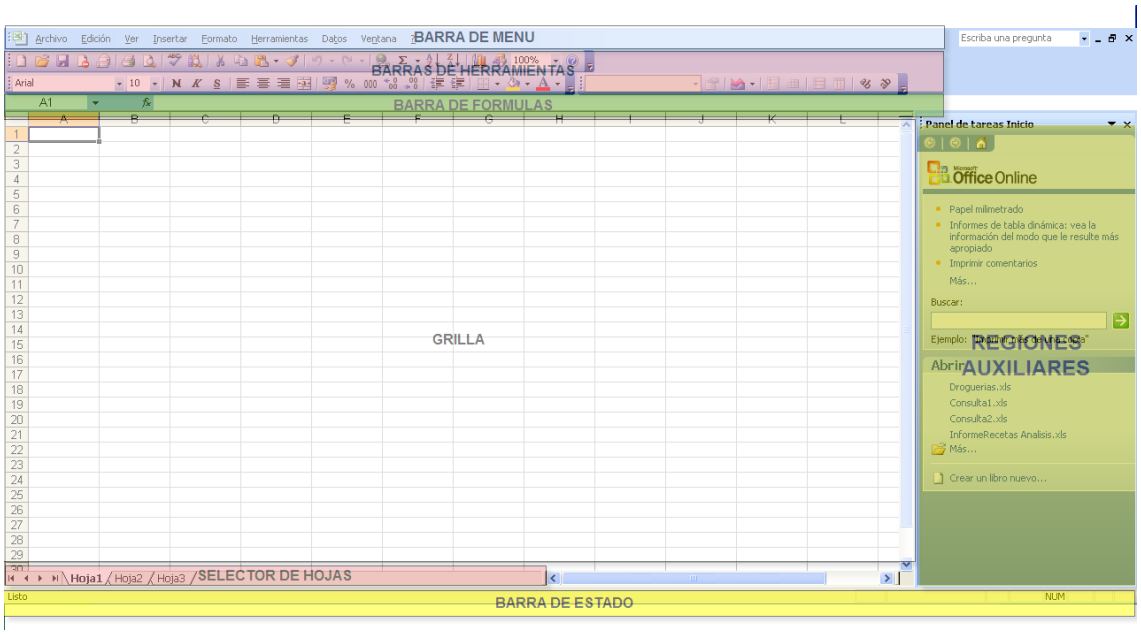

Figura 2 - Componentes Gráficos de Microsoft Excel 2003

En términos generales, estos componentes y su distribución se encuentran prácticamente en todas las aplicaciones relevadas, con pequeñas variantes. No se extenderá en el análisis de las otras aplicaciones ya que no aporta información de importancia para este estudio. Lo que sí debe quedar claro, es que como se describió en requerimientos de la aplicación, el sistema en desarrollo intenta utilizar conceptos y criterios que utiliza Microsoft Excel 2003.

El propósito de realizar una interfaz gráfica similar a Microsoft Excel 2003 es aprovechar la experiencia del usuario con este tipo de aplicaciones que le son de uso diario. De esta forma, la curva de aprendizaje del usuario es mucho menor.

4.1 Diseño de Interfaz de GelSheet

Como línea base del diseño de la aplicación, se ilustrará cómo el área de trabajo es distribuida en el entorno del usuario. Como muestra la Figura 3, el área de trabajo está organizado en cinco regiones: Norte, Sur, Este, Oeste y Centro.

Si bien actualmente no están en uso todas las regiones que se muestran, sí fueron implementadas, y fueron diseñadas para soportar que la aplicación sea extensible y dar la posibilidad de aportar nuevos complementos.

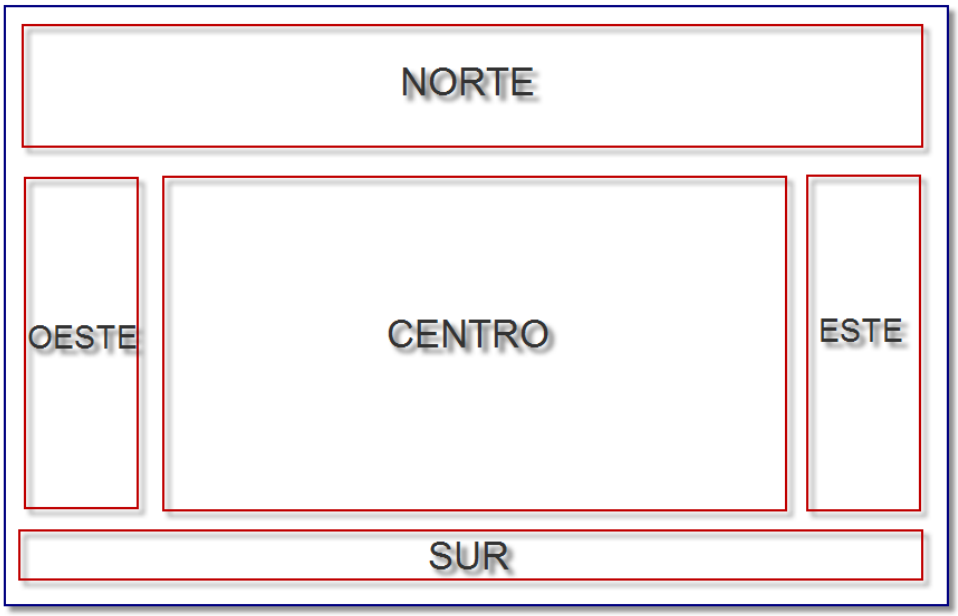

Figura 3 - Distribución del Área de Trabajo en Regiones

Para mantener la distribución que mantiene la aplicación Microsoft Excel, las barras de herramientas y la barra de fórmulas están distribuidas en la región Norte.

La región Centro es utilizada por la grilla. Actualmente dicha región abarca las áreas que contemplan Oeste y Sur aunque son reservadas para uso futuro.

El área abarcada por la región sur está diseñada para desplegar el conjunto de hojas del libro abierto, pero actualmente la aplicación soporta una única hoja por libro, razón por la cuál ha sido ocultada, ya que carece de sentido y ocupa área de trabajo para el usuario.

Concluyendo, el área de trabajo actualmente útil puede resumirse tal como lo muestra la Figura 4 en la región norte y centro.

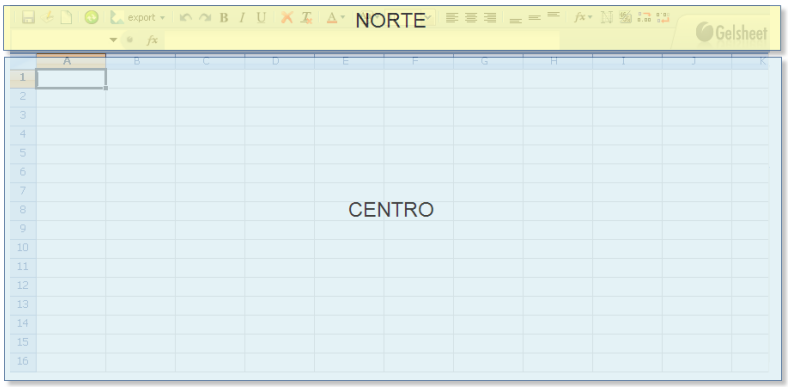

Figura 4 - Área de Trabajo implementada

GelSheet no implementa actualmente una barra de menús, ya que no posee una cantidad suficiente de funcionalidades que ameriten la incorporación de la misma y desperdiciaría espacio en el área de trabajo del usuario.

A grandes rasgos la implementación actual del sistema contempla únicamente las barras de herramientas, barra de fórmula y la grilla.

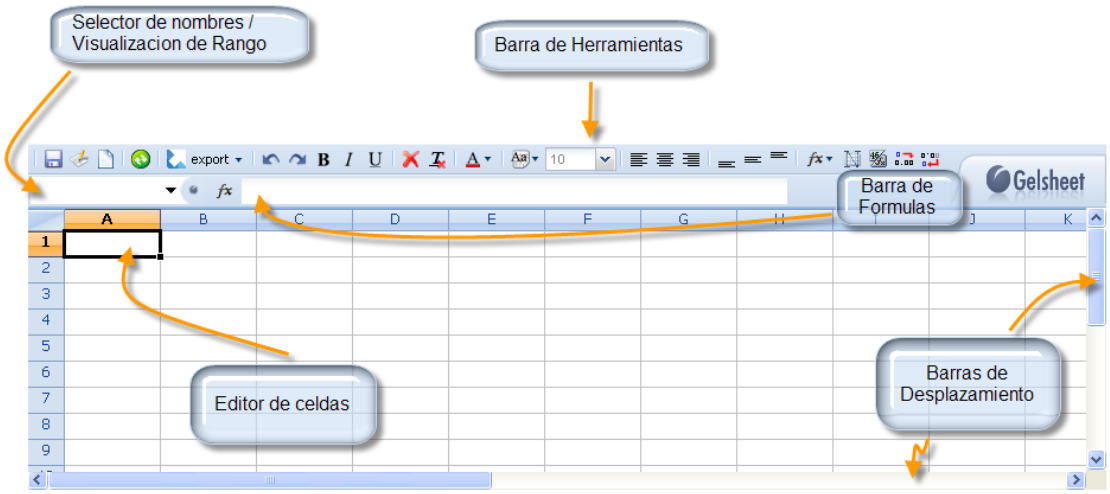

Figura 5 - Componentes Gráficos de GelSheet

La Figura 5 muestra los componentes más relevantes de la interfaz gráfica. En la parte superior (región norte), se encuentran las Barras de Herramientas, la Barra de Fórmulas y el Selector de Nombres/Visualizador de rangos activos. El comportamiento de este último alterna según la operación de usuario que se esté ejecutando, por defecto despliega la posición (dirección) de la celda activa. Si el usuario realiza una selección de celdas, entonces muestra el rango que se está seleccionando. Por otro lado si se desea ver los Nombres de Rangos definidos por el usuario, se puede realizar mediante un clic sobre el Combo. Por otro lado, si se desea agregar un Nombre nuevo, basta con seleccionar el rango y escribir el nombre en el mismo Combo.

Este comportamiento es similar al que Microsoft Excel maneja, con la salvedad que en lugar de desplegar la dirección del rango de la selección, Excel despliega el tamaño de la selección (cantidad de filas y de columnas).

En la región centro se encuentra la Grilla y las barras de desplazamiento (*scrollbars*), estas últimas son consideradas como elementos aparte, ya que tienen un rol importante en la construcción del sistema, por criterios que se describirán más adelante en el capítulo Diseño.

5 ARQUITECTURA DE LA APLICACIÓN

En este capítulo se presenta una descripción de la arquitectura de la aplicación. Seguramente sea de interés para el lector consultar el documento de *"Anexos – Implementación"*, para comprender de qué manera se implementa la arquitectura aquí descripta.

Primero se presenta una visión "tradicional" de la arquitectura del mismo, y posteriormente se verá una descripción de los componentes que integran la solución. Luego se describe de qué manera se distribuyen dichos componentes, así como los protocolos y medios que se utilizan para la comunicación.

Se detallan a grandes rasgos las principales características arquitectónicas de la aplicación, separando la misma en Subsistemas y Componentes.

La aplicación se basa en una arquitectura "Cliente-Servidor", es decir que la aplicación se divide en dos subsistemas principales que son prácticamente independientes, donde uno corre en el cliente y otro en el servidor. Esta arquitectura consiste básicamente en una aplicación cliente interactuando con un servidor, mediante peticiones (13).

Otra decisión importante tomada es que el almacenamiento de datos se realizará completamente en una base de datos. Esta característica permite centralizar toda la información.

El sistema está pensado para un número "ilimitado" de programas clientes que utilizan al servidor de aplicaciones como forma de acceso a la lógica del modelo, siendo este último el que accede a la información, interactuando con el servidor de base de datos, conformando una arquitectura en tres niveles.

Al ser esta una aplicación Web, es el navegador Web es el cliente, y el servidor Web es el servidor de aplicaciones.

5.1 ARQUITECTURA CLIENTE-SERVIDOR

Se describirá a continuación en qué consiste la arquitectura "Cliente-Servidor", y luego se expone el criterio por el cual se ha elegido dicho modelo.

Como se mencionó anteriormente, consiste esencialmente en un grupo de aplicaciones clientes quienes solicitan recursos del servidor, quien asume un rol pasivo en la comunicación limitándose a recibir, procesar y responder las peticiones de los clientes. Es decir, el servidor nunca inicia un diálogo con el cliente, simplemente se encuentra en un estado de escucha como muestra la Figura 6 - Esquema Cliente-Servidor.

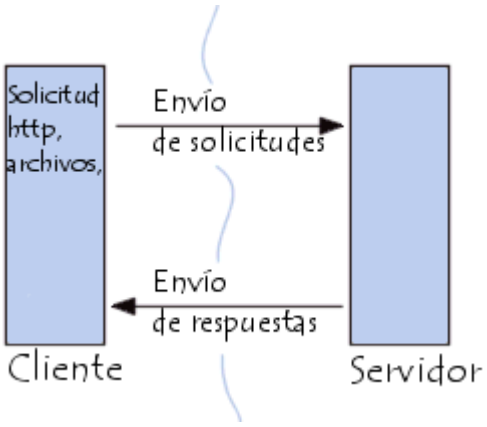

Figura 6 - Esquema Cliente-Servidor

El sistema está diseñado para soportar una arquitectura de tres niveles, es decir que el acceso a datos está desacoplado de la lógica del servidor, con lo cual se agrega un tercer componente de nivel tres, que es el responsable del acceso físico a los datos. De esta forma el sistema queda conformado por tres componentes principales: el cliente, el servidor de aplicaciones y el servidor de bases de datos (nivel 1, 2 y 3 respectivamente). La Figura 7 - Esquema Cliente-Servidor en 3 niveles muestra un bosquejo del modelo.

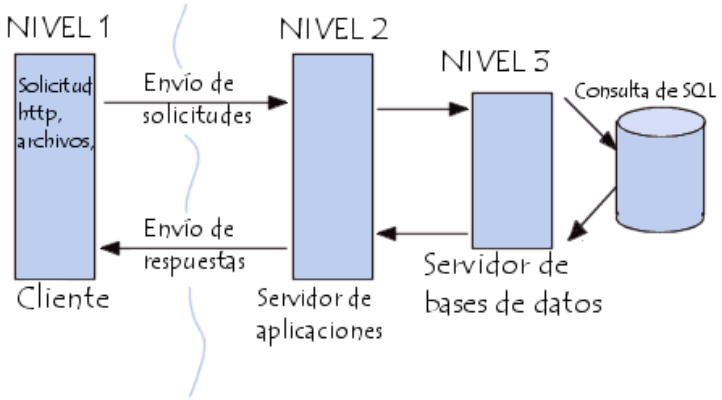

Figura 7 - Esquema Cliente-Servidor en 3 niveles

No es obligatorio que la arquitectura posea tres niveles dado que puede ser reducido a un modelo dos niveles en donde el servidor de aplicaciones y el servidor de bases de datos se encuentran físicamente en el hardware. Este es un escenario típico de las aplicaciones Web tradicionales. La Figura 8 - Esquema Cliente-Servidor en 2 niveles muestra la abstracción descrita.

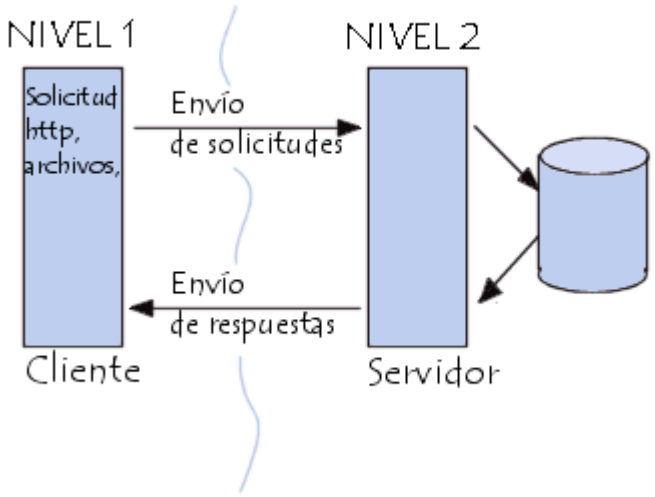

Figura 8 - Esquema Cliente-Servidor en 2 niveles

5.2 Elección del Modelo Arquitectónico

El problema de elección del modelo arquitectónico se basa en la necesidad de construir un sistema constituido por varios consumidores de servicios, sobre una fuente de datos centralizada.

La aplicación debe ser una aplicación Web, que corra sobre un navegador. El navegador únicamente solicita recursos a través de peticiones, pero nunca provee recursos, es exclusivamente un consumidor. No es posible plantear una arquitectura en la cual la solicitud de recursos o el inicio de una comunicación deban ser implementados por otro sistema que no sea el navegador Web.

El patrón arquitectónico que resuelve el problema que se plantea anteriormente es conocido como "Arquitectura Cliente-Servidor", alternativa que sustituye a la arquitectura monolítica en la que no hay distribución de recursos entre sistemas. La Web es modelada mediante Servidores Web, que proveen recursos a los navegadores Web que actúan como consumidores (clientes), y la comunicación nunca es iniciada por el servidor.

En las aplicaciones Web, las operaciones de persistencia, control de acceso de usuarios y de manejo de archivos no pueden ser gestionadas en los navegadores existentes debido a restricciones de diseño de estos.

5.3 Decisiones de Diseño

En las aplicaciones distribuidas, donde varias terminales clientes acceden a un mismo servidor, a medida que crece la cantidad de clientes, tiende a concentrarse la carga en el servidor. El enfoque del diseño arquitectónico se basa en cómo será distribuida la carga de procesamiento de las operaciones de la aplicación entre los distintos sistemas.

Existen dos enfoques opuestos:

1. **Cliente ligero**: aplicaciones clientes tontas consumiendo tareas procesadas en el servidor.
2. **Cliente pesado**: Un servidor dedicado a las operaciones básicas del DBMS y de aquellas que deben centralizar la información, y terminales inteligentes que realizan la mayor parte de las tareas.

Otra alternativa intermedia consiste en balancear la carga de procesamiento entre los distintos sistemas.

Nuestro criterio de diseño, se basa en delegar la mayor cantidad de procesamiento posible en las terminales clientes (cliente pesado).

VENTAJAS:

- **Mayor cantidad de clientes por servidor**: Una carga de procesamiento menor en el servidor, permite destinar recursos para atender a más clientes.
- **Servidor con menor necesidad de recursos**: Esta característica puede ser relevante considerando que existe una gran cantidad de servidores Web comerciales que comparten recursos con varios sistemas, repercutiendo en rendimiento por sistema.
- **Mejor tiempo de respuesta**: Cada petición toma un tiempo de transmisión de datos sumado al tiempo de procesamiento en el servidor. Luego los datos deben ser procesados en el cliente. Esto repercute negativamente en la interacción del usuario con la aplicación cliente. Al existir menor cantidad de solicitudes se mejora la experiencia del usuario con la aplicación.

DESVENTAJAS:

- **Clientes con necesidad de mayores recursos**: Al delegar mayor cantidad de procesamiento a los clientes, se les exige una mayor disponibilidad de recursos. Si bien esta característica puede ser una limitante, las capacidades de procesamiento y de recursos de los computadores actuales es cada vez mayor y más económicas.
- **Mayor transferencia de datos en la carga inicial**: Al adoptar mayores funcionalidades, el cliente deberá descargar mayor cantidad de código para la implementación de las mismas.

Ponderando las ventajas y desventajas del enfoque escogido, se consideró que la alternativa de reducir la carga del servidor delegando la mayoría del procesamiento a cada cliente, mejora el tiempo de respuesta de la aplicación, algo fundamental en aplicaciones interactivas como una Hoja de Cálculo.

Con estas premisas la aplicación servidor será necesariamente el responsable de las siguientes tareas:

- Persistencia: Guardado y carga de las entidades involucradas en la aplicación.
- Gestión de Usuarios: Control de acceso, permisos sobre los recursos, integración con sistemas de terceros.

- Importación/Exportación de Formatos de Archivo: Debido a que los navegadores no permiten el manejo de archivos, las tareas de conversión son delegadas al servidor.

Las responsabilidades del cliente serán las siguientes:

- Ejecutar la Interfaz de Usuario.
- Procesar las operaciones de la lógica de la hoja de cálculo (exceptuando las funcionalidades de exportación e importación).

5.4 SUBSISTEMAS

A continuación se expondrán los dos principales subsistemas que intervienen en la arquitectura. Sin entrar en detalle de la implementación, se describirá al sistema como la abstracción del modelo en dos niveles: "cliente" y "servidor" (*GelSheet-Client* y *GelSheet-Server* respectivamente).

5.4.1 DESCRIPCIÓN GENERAL

Se define *GelSheet-Server* como el sistema compuesto por uno o más servidores Web (servidores de aplicación), sumado al servidor de bases de datos. Por otro lado *GelSheet-Client* es la aplicación que se ejecuta en el navegador Web de cada cliente e interactúa con el servidor Web por medio de peticiones y respuestas HTTP.

La Figura 9 - Visión distribuida de la Arquitectura, muestra cómo se conectan e interactúan todos los componentes involucrados.

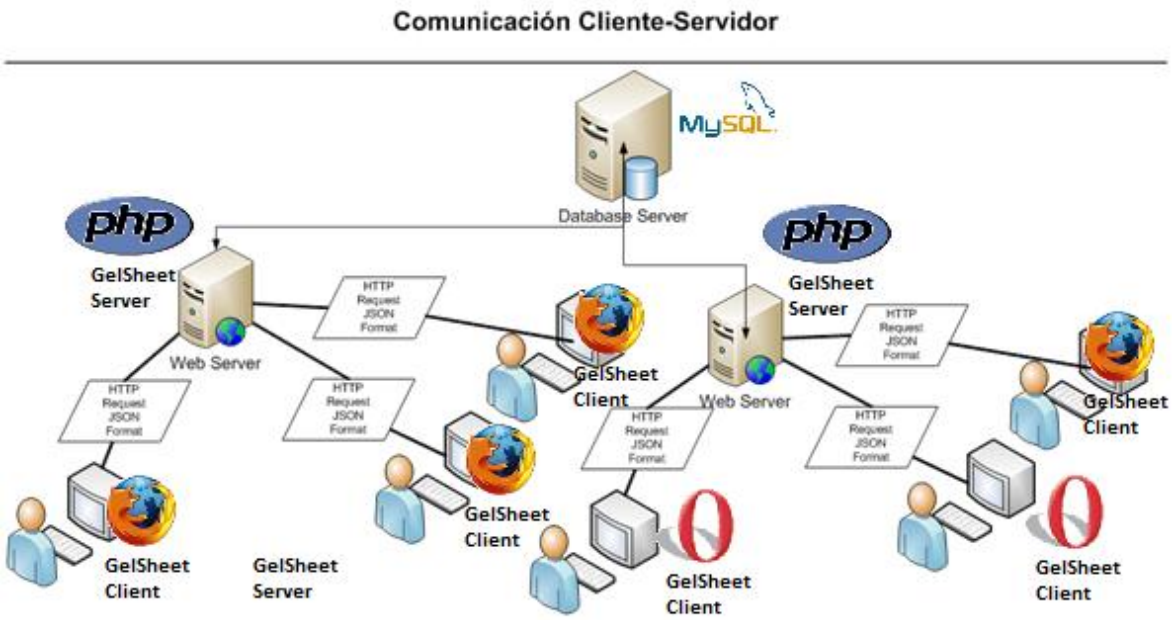

Figura 9 - Visión distribuida de la Arquitectura

La aplicación Cliente implementa todo lo que refiere a la interacción con el usuario directamente, es decir, se encarga de la GUI (interfaz de usuario), y la lógica de las operaciones de la hoja de cálculo. Por otro lado el servidor está encargado fundamentalmente de la persistencia, seguridad, la gestión de documentos (importación/exportación) y el control de concurrencia (a futuro). En términos generales se encarga del procesamiento de datos y su persistencia.

Estos subsistemas son descritos más detalladamente en la documentación de implementación pero serán presentados aquí, a grandes rasgos, las principales características.

5.4.2 GELSHEET-SERVER

A continuación se describen las principales responsabilidades del servidor

- Persistencia:
    - El almacenamiento de datos se realiza completamente en una base de datos, por lo que es responsabilidad del servidor encargarse de este aspecto. Esto involucra todo lo relacionado al guardado y la carga de las hojas de cálculo así como los datos referentes a idiomas y configuraciones varias.
- Lógica:
    - Seguridad: Controles de acceso a los datos. Se desarrolló una API para permitir más fácilmente la integración con otros sistemas.
    - Exportación e Importación de documentos en distintos formatos.
    - Integración con sistemas externos (Implementación actual: *OpenGoo*).
    - Comunicación con el motor de base de datos (persistencia) y con el navegador Web (interfaz de usuario).
    - Se provee una API que permite ejecutar las operaciones del sistema vía parámetros HTTP para facilitar la integración e interacción con el cliente.

5.4.3 GELSHEET-CLIENT

- Interfaz de Usuario:
    - Se ejecuta completamente en el navegador Web.
    - Interactúa directamente con el usuario y le provee una interfaz de teclado y mouse para la utilización de la herramienta.
- Lógica:
    - Implementa el corazón de la lógica de la hoja de cálculo, siendo implementadas en este nivel la mayoría de las funcionalidades vinculadas al

tratamiento de la hoja de cálculo. Delega al servidor la persistencia y la exportación e importación de documentos.

- Comunicación con el servidor y sistemas externos (*OpenGoo*).

5.5 CONCLUSIONES

El modelo arquitectónico adoptado es "Cliente-Servidor", caracterizado por una aplicación cliente que posee la mayor parte de la lógica del sistema, siendo el módulo servidor esencialmente responsable de la persistencia y conversión entre formatos de archivo. A este tipo de distribución de carga suele denominarse "Cliente Pesado", y los principales criterios de su elección se basan en la interactividad con el usuario y la liberación de carga al servidor Web.

6 DISEÑO

En este capítulo se exponen las decisiones de diseño de la aplicación desarrollada. Dada la decisión arquitectónica tomada de dividir el sistema en dos grandes componentes desacoplados (cliente y servidor), se agrupan las decisiones de diseño en tres clases: una clase general donde se especifican temas comunes y de comunicación entre ambos componentes, luego serán tratadas las decisiones específicas del servidor y finalmente las referentes a la aplicación cliente.

6.1 CONCEPTOS GENERALES

6.1.1 RENDIMIENTO

Tanto a nivel de cliente como del servidor se optó por modularizar los componentes en múltiples archivos.

Al ser un sistema Web, para cargar la aplicación cliente, esta es solicitada al servidor cada vez que se inicia. Por tal razón la decisión de dividir los recursos en diversos archivos puede repercutir en el rendimiento negativamente tanto para el cliente como para el servidor.

El cliente se ve perjudicado principalmente en el tiempo de carga de la aplicación ya que debe solicitar cada uno de esos recursos individualmente, sincronizarlos y luego procesarlos. El esfuerzo que requiere es mayor que para solicitar un único recurso.

Por otro lado el servidor se ve afectado al tener que mantener las conexiones para estas solicitudes por cada cliente.

Para mejorar este aspecto se utilizaron dos técnicas: La primera de ellas se encarga de crear un único archivo Javascript a partir de un conjunto de ellos (unificado). Luego dicho archivo pasa por un segundo filtro que se encarga de reducir el tamaño del mismo (minificado). Además esto va acompañado de un caché a nivel del servidor para reducir la carga del mismo debido al procesamiento extra generado por las técnicas anteriores.

UNIFICADO

Se realiza en el servidor cuando se le solicita el archivo "*index.php*". En lugar de escribir en el documento HTML todas las referencias a archivos Javascript externos, este realiza una simple iteración sobre una lista ordenada de archivos Javascript a incluir, pegando su contenido de forma secuencial uno a continuación del otro, creando un único archivo código fuente. Este código generado, se pasa como parámetro a un segundo filtro (minificado) que se encargará de reducir su tamaño.

MINIFICADO

También se realiza del lado del servidor, y su función principal es reducir el tamaño de un determinado código Javascript. Esta técnica consiste en eliminar líneas de código innecesarias como ser: espacios en blanco, saltos de líneas y comentarios, acompañado de renombre de funciones y variables. Como resultado retorna un código prácticamente ilegible para el desarrollador, pero con una reducción considerable del tamaño total. Para la

implementación de esta técnica es utilizada una librería PHP, llamada "*js-min*" (14), la cual provee dicha funcionalidad, además de otros comportamientos un poco más complejos.

CACHÉ

El resultado de los procesos de unificado y minificado del código se almacenan en un único archivo llamado "gelsheet-min.js". Con esta técnica se reducen considerablemente los tiempos de carga de la aplicación y el volumen de información transmitida.

6.1.2 INTERNACIONALIZACIÓN

Uno de los requerimientos de la aplicación es que soporte una interfaz con múltiple-idioma. La definición del idioma de la aplicación se encuentra en el archivo de configuración principal, pero también es posible sobrescribir este valor con un parámetro por URL con lo cual facilita la integración con otros sistemas. La aplicación cliente consume un recurso elaborado por el servidor con estas configuraciones.

La tabla de traducciones se encuentra en un único archivo de configuración en el módulo del servidor. Esta se basa en un modelo de diccionario "*Key-Value*" (Clave-Valor). Se define una frase clave y para cada idioma se asocia con un valor (la traducción del texto en el idioma correspondiente).

6.1.3 SEGURIDAD

Como todo sistema Cliente-Servidor, el control de acceso y aspectos de la seguridad son principalmente manejados a nivel del servidor, ya que la aplicación cliente es fácilmente vulnerable, debido a que cualquier sujeto puede alterar el código.

Por esta razón las funcionalidades de seguridad son efectuadas en el servidor, y como el producto debe ser integrable a sistemas de terceros, debe ser manejada de forma desacoplada. Esto se explicará mejor en la siguiente sección, en la cual se describe el diseño de la aplicación servidor.

6.2 APLICACIÓN SERVIDOR

Para implementar la recepción y procesamiento de los mensajes del lado del servidor, se optó por una solución simple, patrón de diseño "*Front Controller*". Este modelo es popular en aplicaciones Web de gran porte, por ejemplo los CMS (*Content Management System*).

6.2.1 PATRÓN FRONT CONTROLLER

Es un patrón de diseño muy utilizado en varios lenguajes de programación y está muy relacionado al diseño/desarrollo de aplicaciones Web (15). Se basa principalmente en tener un único punto de entrada accesible por las peticiones HTTP realizadas por el navegador, de forma tal de centralizar todos los accesos y permitir un mayor control de la aplicación así como la reutilización de código (16).

Básicamente nuestra estructura lógica consiste en un conjunto de Controladores organizados de forma jerárquica, de forma tal que sólo un subconjunto de ellos pueda ser invocado desde

la interfaz Web, es decir aquellos que sean sub-clase de "FrontController". Estos se encargan tanto de procesar algunos aspectos básicos relacionados a la seguridad y controles de acceso, como resolver o delegar a otros controladores el procesamiento de una tarea invocada desde afuera. Para ello fue necesario definir una serie de convenciones para los parámetros *POST* y *GET* de las peticiones Web.

6.2.2 SEGURIDAD

Las características de seguridad de la aplicación dependen de las políticas de acceso a los recursos del sistema en dónde se encuentre alojado. Cabe decir que los criterios de acceso pueden variar de un sistema a otro y que el sistema implementado se debe adaptar a las políticas del sistema anfitrión.

Por esta razón fue necesario implementar un modelo abstracto en el cual se contemplen los niveles de acceso a los recursos, sin tener en cuenta cómo son implementados.

Esto se logra delegando a la aplicación anfitriona la responsabilidad de decidir si un sujeto tiene permisos para realizar cierta operación sobre determinados objetos.

Para ilustrar estos conceptos, son expuestos los tres elementos que componen las políticas de seguridad de GelSheet dentro de OpenGoo.

- Sujetos: Usuarios.
- Objetos: Libros.
- Operaciones: Crear, Leer, Editar y Borrar.

Previo a la ejecución de una operación, GelSheet invoca al módulo de seguridad de OpenGoo para que decida si es autorizada o no. Luego, en caso afirmativo, ejecuta la operación o retorna un error en caso contrario.

Los detalles y especificaciones técnicas son abordados en el *"Anexo – Implementación"* en la sección Seguridad.

6.3 APLICACIÓN CLIENTE

La aplicación cliente es responsable de implementar la interfaz de usuario (GUI), así como la lógica de las operaciones de la hoja de cálculo. La persistencia es delegada al servidor, así como las operaciones de conversión de formatos de archivo (importación/exportación), la seguridad y las configuraciones de usuario, idioma, etc.

El aspecto más importante del diseño refiere a la cantidad de información que debe soportar la aplicación, ya que el objetivo de la misma es el soporte de gran cantidad de celdas por hoja de cálculo.

Un volumen de datos grandes se traduce en un mayor tiempo de procesamiento de los mismos, que deriva en un consumo de recursos que puede ser excesivo. Por esta razón algunos navegadores Web poseen una característica de detección de bucles largos en la ejecución del código que permite al usuario interrumpir el código cuando esto sucede. Este comportamiento pretende proteger al usuario ya que el navegador es capaz de alertarlo en

situaciones que un código (eventualmente malicioso) se ejecute y consuma los recursos del computador.

Es claro que existen aplicaciones que requieren necesariamente la ejecución de estos bucles, y lo que se presenta como una ventaja para el usuario, termina siendo un inconveniente para el desarrollador. Por lo tanto, al momento de desarrollar la aplicación, es necesario tener en cuenta esta característica. La principal decisión de diseño en la aplicación cliente, estuvo relacionada con la inicialización de la grilla, y su solución es expuesta a continuación.

6.3.1 LA GRILLA

El corazón de la interfaz se basa en la visualización de la hoja de cálculo, que es vista como una grilla con filas y columnas etiquetadas. Esta deberá soportar una gran cantidad de filas y columnas sin disminuir sensiblemente el rendimiento.

Resulta intuitivo el concepto de crear e inicializar una grilla con la cantidad de filas y columnas que la hoja requiera, sin embargo surgen limitaciones en la implementación por dos razones: primero, el tiempo que puede tomar en generar una cantidad grande de celdas atenta con los tiempos de respuesta aceptables para el usuario. La segunda limitante es añadida por los navegadores tradicionales, que como se mencionó previamente, detectan bucles largos y dan la opción al usuario de detener la ejecución[4].

La solución que utilizan las aplicaciones Web relevadas, se enfoca en inicializar una grilla con un volumen de celdas prudente para el cual no sea detectado como un bucle largo por el navegador (con un rango que no excede de 100 a 500 filas por 20 a 25 columnas). Como contrapartida cuando el usuario necesita más celdas debe solicitar a la aplicación, a través de un comando, que genere más filas o más columnas, acción que a su vez genera una cantidad de celdas también acotada. Esta solución resulta efectiva para planillas con cantidad de filas y columnas pequeñas, pero resultan bastante tediosas para aquellos usuarios que utilicen planillas con un gran volumen de información.

La solución elegida se basó en hacer un mapeo de celdas visibles (*VCells*) con las celdas lógicas (*Cells*), en el cual la grilla tendría un número pequeño de celdas (únicamente las que son visibles en la pantalla). El valor y el formato de cada celda se ajustarían según el valor de la celda lógica que esté asociada.

La Figura 10 muestra una representación gráfica del modelo descrito anteriormente.

[4] Este problema se presentó por primera vez en el desarrollo del primer prototipo de la aplicación, mientras se intentaba inicializar una grilla con un número mayor a mil quinientas celdas aproximadamente

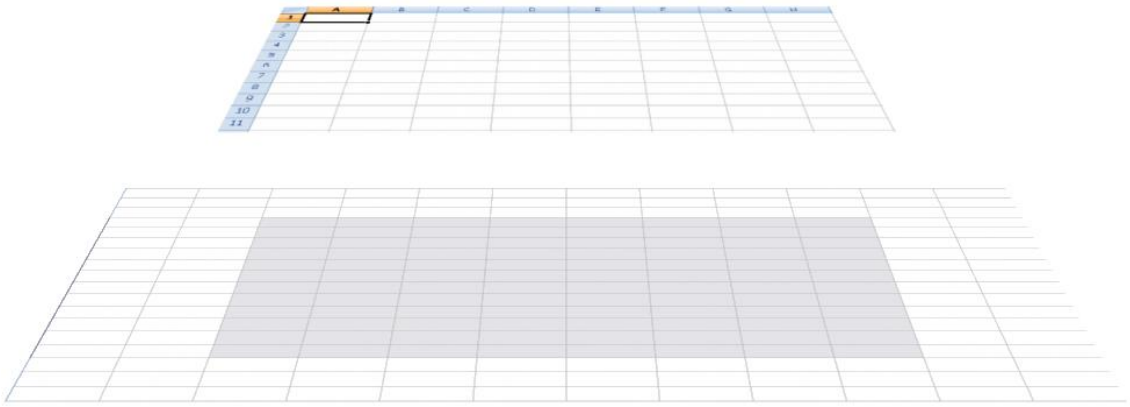

Figura 10 - Mapeo de Celdas Visuales (VCells) con Celdas Lógicas (Cells)

De esta manera se soluciona el problema de la representación de la matriz de celdas en pantalla (eventualmente ilimitada), pero generando una mayor complejidad en la lógica de la aplicación. De hecho esto último generó una complejidad grande en el desarrollo de la aplicación, ya que contemplar el mapeo de cada celda exigía también la implementación de las barras de desplazamiento propias, y un control riguroso de cada actualización de contenido, tanto de celdas como filas y columnas, al igual que para los formatos de los mismos.

Para dar la sensación al usuario de que realmente se desplaza por la hoja, se generaron eventos que son disparados cuando la región visible se mueve de posición. Una vez que esto sucede, el evento es capturado y el contenido de todas las celdas visibles es refrescado tanto a nivel del contenido, como del formato y del tamaño.

Por otro lado existen situaciones en las cuales el valor de una celda, afecta el valor de otras, por ejemplo en fórmulas que utilizan el valor de otras celdas como referencia. Una vez que las celdas referenciadas cambian, también debe cambiar el valor del resultado de la fórmula.

Para lograr que el cambio del contenido de una celda sea visible automáticamente, se utiliza una técnica similar, se genera un evento cada vez que una celda es actualizada, que a su vez es capturado y es disparado el procedimiento de refresco.

El procedimiento de refresco de las celdas visibles, es utilizado en diversas situaciones, tales como el cambio en el ancho y alto de columnas y filas respectivamente. Una vez que el tamaño es alterado, también es invocado el procedimiento de refresco.

6.3.2 Estructura interna de la Hoja de Cálculo

La aplicación divide los componentes visuales de los componentes de la lógica de la hoja de cálculo. Como se mencionó anteriormente, el valor de cada celda se almacena en celdas lógicas, las cuales están contenidas en una matriz dispersa de objetos.

La decisión de utilizar una matriz dispersa se basó en los siguientes fundamentos:

- En general una hoja de cálculo contiene una gran cantidad de celdas vacías, en relación a la cantidad de celdas disponibles.

- Es una estructura incremental, basada en el uso eficiente de los recursos, considerando que la reserva de memoria es a demanda.
- Javascript posee de forma nativa arreglos asociativos dispersos (*Sparse Arrays*).

La matriz de celdas consiste en un arreglo bidimensional con la estructura de almacenamiento *List of Lists* (LIL).

Cabe destacar que si bien la aplicación no soporta a nivel de la interfaz de usuario múltiples hojas por libro, la lógica implementa libros con múltiples hojas, las cuales contienen la matriz de celdas.

6.4 CONCLUSIONES

El servidor está basado en un modelo "Front Controller", el cual provee un acceso centralizado a las peticiones Web del navegador.

Las políticas de seguridad del sistema son implementadas a nivel del servidor y están sujetas a las políticas del sistema anfitrión, por lo tanto se ha diseñado de forma abstracta un módulo que debe ser implementado para cada aplicación anfitriona.

La aplicación cliente realiza una carga de recursos al iniciarse, dentro de los cuales se encuentran los códigos fuentes. Para optimizar el tiempo de carga se optó por el unificado de todos los fuentes en un único archivo (minificado).

La principal decisión de diseño sobre la aplicación cliente se centra en la estructura de datos para mantener en memoria la hoja de cálculo. La estructura escogida es una matriz dispersa fundamentada en la baja relación de celdas ocupadas en función de las celdas disponibles. Esta implementación está complementada por un mapeo de un conjunto pequeño de celdas visibles sobre un universo grande de celdas lógicas con el fin de evitar pérdida de rendimiento y espacio en memoria del navegador. Un consumo de recursos excesivo por la aplicación puede derivar en que el navegador alerte al usuario de dicha situación, dando la posibilidad de interrumpir la ejecución.

7 IMPLEMENTACIÓN

En esta parte del documento se describirá los aspectos generales de la implementación. Para profundizar en los aspectos más específicos (como criterios de implementación, descripción de los modelos, nomenclatura, especificaciones técnicas, así como características de la integración con OpenGoo) se recomienda leer el documento de Anexos sección Implementación.

La aplicación fue desarrollada bajo un diseño del patrón Cliente-Servidor utilizando los lenguajes de programación Javascript y PHP (17) respectivamente. La comunicación entre componentes es realizada utilizando la tecnología AJAX. La persistencia es gestionada por una base de datos en MySQL (18).

Conceptualmente el cliente y el servidor son considerados como dos sistemas distintos e independientes comunicándose entre sí, por lo que son abarcados de forma separada a cada uno. Una característica destacable que ofrece el producto es que la aplicación Cliente puede ser utilizada independientemente (excluyendo la persistencia y configuraciones del usuario). De la misma forma la aplicación Servidor ofrece servicios que pueden ser consumidos por otra aplicación totalmente distinta.

7.1 APLICACIÓN CLIENTE

La aplicación Cliente es una aplicación Web, es decir que corre sobre un navegador, y se implementó utilizando las tecnologías de Javascript y HTML. La comunicación de mensajes con el servidor se basó en el conjunto de tecnologías que abarca AJAX, esto es peticiones HTTP asíncronas con formato de mensajes en JSON (19).

El desarrollo de la interfaz gráfica hace uso de la librería ExtJS (20), también utilizada por OpenGoo, implementada en Javascript.

7.1.1 PARADIGMA DE DESARROLLO

El modelo de implementación se basó en el Modelado de Orientación a Objetos (21). Si bien Javascript no es un lenguaje totalmente orientado a objetos es posible mediante algunas técnicas simular un comportamiento de Orientación a Objetos (OO).

Javascript no soporta herencia, ni definición de interfaces, pero sin embargo es posible simular algunos de estos comportamientos mediante técnicas de implementación, las cuales se describen en el "*Anexo – Implementación"*.

El motivo principal por el que se ha utilizado el paradigma de la Orientación a Objetos, obedece a que fomenta la reutilización y extensión del código. Permite a su vez, una abstracción de componentes, los que pueden ser re-implementados e integrados al sistema eventualmente sin cambios en otros componentes.

Una desventaja que introduce la OO, es que en general se pierde eficiencia y gastos de procesamiento e invocaciones a operaciones que en otros modelos de programación pueden ser evitados.

No se pretende abarcar en este documento el modelo de Orientación a Objetos, la intención es justificar brevemente la elección de este paradigma. Debe quedar claro entonces, que el criterio principal ha sido la extensibilidad y la modularidad que permite esta metodología.

7.1.2 COMPONENTES PRINCIPALES

Como la aplicación cliente implementa toda la interfaz gráfica y la lógica interna de las operaciones de la hoja de cálculo, los principales componentes se dividen en dos tipos: los referentes a la GUI (*Graphic User Interface*) y los referentes a la lógica de las operaciones de la hoja de cálculo.

7.1.3 COMPONENTES DE LA GUI

Cualquier grilla de este tipo de aplicaciones tiene dos regiones, situadas en la parte superior y en la parte izquierda que son las etiquetas de las columnas y filas respectivamente. Estas áreas permiten identificar en qué dirección se encuentra cada celda, y en particular, para la celda activa, se colorea la posición de la columna y fila a la que pertenece para identificarla visualmente. Estas regiones son utilizadas también para cambiar el tamaño, y seleccionar las filas y columnas según corresponda.

Entrando en detalle, surgen componentes de la grilla que forman parte de la edición y selección del contenido, si se observa la Figura 5 se puede ver el editor de celdas, elemento que existe en todas las aplicaciones de hojas de cálculo. El editor de celdas es un componente (junto con la barra de fórmulas) que permite editar el contenido de la celda activa y/o conjunto de celdas seleccionadas. El editor de celdas no es parte de cada celda, sino que se trata de un componente separado que se posiciona y ajusta en la celda activa.

Otro elemento destacable de la grilla son las barras de desplazamiento, que a diferencia de una implementación convencional, éstas no resultarían otra cosa que parte de un control existente. Sin embargo, como se ha descrito anteriormente, el funcionamiento de la grilla consiste en un mapeo de celdas visuales con celdas lógicas, lo cual imposibilita utilizar las barras de desplazamiento convencionales de forma estándar. Para simular el comportamiento de este elemento, se creó un componente adicional que encierra a la grilla y que posee un área de tamaño acorde a las dimensiones de las celdas lógicas editadas. Este último componente es llamado "*ScrollBars*". Cuando el usuario se desplaza, se captura el evento de manera de refrescar el contenido de las celdas visuales acorde al contenido de las lógicas.

Otros componentes de la interfaz de usuario son las barras de herramientas, la barra de fórmulas, los cuadros de diálogo, y el selector de nombres. Los mismos son implementados utilizando la librería ExtJS, dado que provee herramientas y componentes gráficos, fácilmente configurables y adaptables.

7.1.4 LÓGICA

A continuación se detallarán algunos conceptos que, a nuestro criterio resulta conveniente destacar acerca de la lógica de la aplicación.

Comunicación mediante Eventos

La comunicación entre objetos de distintos componentes se realizó mediante registros y disparo de eventos. Para los objetos de distintas capas se optó por el modelado de eventos, de esta manera es posible mantener un bajo acoplamiento entre los distintos componentes.

Fórmulas

Una vez que se ingresa una fórmula en una celda el sistema realiza las siguientes acciones:

1. Descomposición:
   Como primer paso es necesario identificar los símbolos involucrados en las fórmulas (direcciones, funciones, argumentos, nombres, operadores).
   Para descomponer en componentes las fórmulas se utilizó una librería externa implementada en Javascript. Esta recibe como argumentos una cadena de caracteres y retorna como resultado una pila de símbolos (*tokens*).
2. Validación Sintáctica:
   Una vez descompuesta la fórmula se realiza la validación sintáctica, en caso de no ser válida se retorna un error.
3. Traducción de Nombres:
   Una vez validada la sintaxis de la fórmula en elementos identificables, se realiza el mapeo de nombres y rangos a direcciones de celdas, junto con el chequeo de nombres válidos.
4. Chequeo de Referencias Circulares:
   Previo al cálculo de la fórmula se realiza la comprobación de referencias circulares, esto es, que una fórmula no contenga referencias de celdas que son referidas dentro de las celdas que se encuentren en la misma (que no exista un ciclo en las referencias).
5. Cálculo:
   Luego de las traducciones y validaciones, se procesa la fórmula aritméticamente, obteniendo un resultado que es almacenado en la hoja.
6. Refresco:
   Una vez obtenido el resultado (eventualmente un error), se invoca al procedimiento de refresco de la grilla para que actualice los resultados en pantalla.

Hacer/Deshacer (*Undo/Redo*)

Para poder implementar esta funcionalidad se utiliza una pila de datos que almacena las transacciones. Cada transacción posee una lista de estados. Estos estados contienen la información de qué operación fue realizada, sobre qué objeto, el valor previo a su ejecución y el nuevo valor.

Las operaciones que permiten guardar y restaurar estados deben soportar un parámetro que indique si debe guardar el estado o no. Cuando se la utiliza de forma estándar la operación guarda su estado. Cuando se realiza el *Undo/Redo*, debe pasarse el argumento de no guardar el estado.

Cada acción de hacer puede repercutir en varios cambios (por ejemplo un cambio sobre una selección), es por esta razón que antes de invocar a las operaciones debe iniciarse una nueva transacción. Las operaciones de hacer/deshacer operan sobre los estados de una transacción.

Una característica de nuestra implementación, es que acepta un número ilimitado de acciones, por lo que es posible deshacer cualquier cambio realizado en la hoja de cálculo, a partir de la carga inicial del documento. La única limitación existente está sujeta a la capacidad de recursos del navegador.

Manejo de Estilos

Cada celda lógica posee la información de su estilo asociado. Dicho estilo maneja información como ser: tipografía, color, alineación, etc. Dado que en las hojas de cálculo es muy frecuente que se repitan estilos entre celdas, se optó por utilizar un manejador de estilo general al libro, generando un identificador por cada estilo distinto. Por lo tanto, las celdas almacenan únicamente el identificador asociado al estilo, ahorrando un espacio considerable en memoria. A su vez, esto permite administrar estilos globales (aunque esta funcionalidad no está implementada actualmente).

7.2 Aplicación Servidor

Anteriormente se ha explicado las características de la aplicación cliente, que es responsable de todas las operaciones de usuario sobre la hoja de cálculo y de los cálculos realizados por la misma. El servidor, por otro lado, es responsable (aparte de la funcionalidad implícita de Servidor Web como responder a las peticiones HTTP), de la implementación de la persistencia, los controles de acceso y las conversiones entre formatos de archivo.

Las tecnologías base utilizadas en la aplicación servidor son PHP como lenguaje de *scripting* y MySQL como motor de base de datos.

En la implementación se mantuvo el enfoque de la programación orientada a objetos, basado en el patrón de diseño "*Front Controller*", junto con una implementación de un ORM (*Object Relational Mapping*) sencillo para el manejo de la persistencia.

Como se ha mencionado anteriormente, el formato de mensajes que se transmiten entre cliente y servidor se realiza mediante JSON (*Javascript Object Notation*).

7.3 Estrategia del Plan de Pruebas

En la presente sección se describen las estrategias y pautas generales elaboradas para el plan de pruebas (*testing*) de la aplicación tanto de forma "*Stand Alone"*, como en su integración con OpenGoo.

Los casos de prueba y los conceptos técnicos, no son incluidos en este apartado. Si este aspecto es de interés para el lector, los mismos son detallados en el "*Anexo - Casos de Prueba"*.

Se realizaron para la verificación un conjunto de casos de prueba de las funcionalidades más frecuentemente utilizadas.

El enfoque planteado para la construcción de los casos de prueba es de caja negra, es decir, se enumeran las funcionalidades las cuales se desea probar y se proporcionan los datos de entrada para corroborar que las salidas o resultados de la interacción con la aplicación sean las esperadas.

Se discriminan las pruebas en dos casos particulares:

- Aquellas pruebas que involucran exclusivamente la implementación de la aplicación a desarrollar.
- Las pruebas que involucran la interacción con el sistema OpenGoo al cual se integra la aplicación.

Las pruebas de caja blanca si bien fueron consideradas durante la realización de las pruebas, resultaron descartadas debido a varias razones que se describen a continuación.

En primer lugar las pruebas de caja blanca hubiesen insumido un tiempo considerable, tanto para el desarrollo de las pruebas como para la elaboración de las mismas, lo que no se consideró viable.

En segundo lugar, se decidió que el plan de pruebas se enfocara exclusivamente a la obtención de resultados visibles para el usuario, por lo que la selección de un enfoque orientado a pruebas de caja negra resulta más adecuada. Ver "*Anexo – Casos de Prueba"*.

7.4 CONCLUSIONES

La aplicación cliente está implementada en HTML y Javascript, se comunica con el servidor mediante AJAX, es decir peticiones asíncronas con mensajes en formato JSON. El servidor fue implementado en PHP y utiliza como motor de base datos a MySQL.

El sistema en su conjunto utiliza el paradigma de desarrollo orientado a objetos tanto en el servidor como en el cliente. Si bien PHP tiene soporte nativo a dicha metodología, Javascript no lo tiene en su totalidad, debiéndose aplicar técnicas para simular ciertos aspectos.

Como aspecto destacable de la aplicación cliente, se ha implementado las funcionalidades de hacer/deshacer las acciones del usuario.

8 CONCLUSIONES Y TRABAJO FUTURO

Al comienzo de este capítulo se describen las conclusiones del proyecto, se realiza una evaluación del proceso y del producto obtenido, exponiendo las fortalezas y carencias del mismo. Finalmente en "Trabajo Futuro", se proponen características y funcionalidades para obtener una aplicación mejorada y más completa.

8.1 CONCLUSIONES

El área de las aplicaciones Web2.0 es un campo en actual proceso de expansión, en particular las *Web Suites*. Últimamente se comienzan a divisar productos que surgen como alternativas serias a sus pares de escritorio, debido a las ventajas que aporta la Web para productos de edición colaborativa. Por otro lado, el desarrollo de este tipo de aplicaciones Web se torna más complejo si se pretende lograr la reproducción fiel de las funcionalidades ofrecidas por aplicaciones similares de escritorio.

En el estudio del Estado del Arte, se concluyó que existen pocas aplicaciones de hojas de cálculo con características similares a las de escritorio. En particular aquellas aplicaciones *Open Source* no lograban satisfacer las funcionalidades mínimas para un editor de hojas de cálculo competente.

Las principales dificultades encontradas se concentraron en el desarrollo de la interfaz de usuario, la que requirió un complejo manejo de sus componentes gráficos e interacción con el usuario. Estas dificultades se acentuaron por los problemas de compatibilidad entre navegadores y sus problemas de rendimiento.

Crear una aplicación integrada a la *suite OpenGoo* que al mismo tiempo pueda ser módulo para otros productos, exigió elaborar soluciones abstractas que mantengan un grado de acoplamiento mínimo. Esto añadió una complejidad adicional tanto en el diseño como en la implementación de la aplicación.

El resultado de este proyecto derivó en un producto que implementa las funcionalidades básicas de manera eficiente y con una rica interfaz de usuario. Si bien no se logró un producto que pueda competir con sus pares comerciales, se logró implementar una plataforma base que puede ser fácilmente extendida por parte de la comunidad.

El cliente quedó conforme con la calidad del producto en general, aunque la aplicación carece de algunas funcionalidades para incorporarla a Feng Office.

8.2 TRABAJO FUTURO

A continuación serán descritos los aspectos que son considerados fundamentales para lograr un producto que compita con sus pares, que debido a la escasez de tiempo, no pudieron ser implementados en esta etapa.

- Compatibilidad de Navegadores.
- Importación de documentos.
- Múltiples hojas por libro.
- Concurrencia.
- Gráficas.
- Extensión de Funcionalidades: copiar, pegar, agregado de nuevas funciones.
- Auditoría de Seguridad.

Compatibilidad entre Navegadores

El producto final es compatible con Mozilla Firefox 3.0+ y Opera 10.0+.

La compatibilidad con Microsoft Internet Explorer 7.0+ es parcial, debido a problemas de rendimiento con este navegador. Si bien se le dedicó un tiempo considerable a este aspecto, quedó pendiente la corrección de algunos *bugs* que aseguren su correcto funcionamiento.

Importación de Documentos

No es posible la edición de documentos provenientes de otras aplicaciones. Para lograr esto, se requiere el desarrollo completo de un módulo de importación. Este punto fue acordado con el cliente, debido a la complejidad de su implementación y la escasez de herramientas en el mercado en las cuales basarnos.

Varias Hojas por Libro

Actualmente la aplicación soporta una hoja por libro. Aunque el modelo contempla esta característica, no está implementado a nivel de la interfaz de usuario, ya que comprometía la robustez del producto y añadía complejidad en el manejo de referencias y fórmulas.

Concurrencia

Es posible la lectura simultánea de las hojas de cálculo, pero la edición concurrente no está completamente implementada. No se retroalimenta a los otros usuarios en línea de los cambios realizados, ni se realiza bloqueos de celdas para evitar pérdida de datos. Dicha situación ocurre cuando el documento es editado simultáneamente, sobrescribiendo los cambios de otro usuario.

Gráficas y Otros Objetos

Es una característica muy utilizada en este tipo de aplicaciones, que debe ser incorporada futuras versiones.

Otras Funcionalidades

Aumentar las cantidades de funciones de la aplicación, tanto matemáticas como estadísticas, y agregar más categorías de funciones. Es posible extender fácilmente el conjunto de operaciones ya que el modelo, por medio de una API en Javascript, provee una sencilla forma de extenderla.

La aplicación debe tener las funcionalidades de portapapeles (copiar, cortar y pegar).

Se debe corregir el conjunto de errores (*bugs*) conocidos, ver *"Anexo – Bugs y Comportamientos no Convencionales"*.

Realización de Pruebas

Es necesario realizar pruebas de la aplicación con usuarios finales para testear las funcionalidades desarrolladas así como para relevar nuevas funcionalidades. De la misma forma, se deben utilizar equipos con diferentes capacidades, para probar el correcto funcionamiento de la aplicación. Este tipo de pruebas serían utilizadas para verificar las decisiones tomadas en cuanto al diseño del producto y evaluar la necesidad de realizar cambios.

Auditoría de Seguridad

Es necesario realizar un estudio más exhaustivo para validar que la aplicación sea robusta ante ataques frecuentes en la Web como ser: *Cross-Site Scripting* (XSS) y *SQL Injection*, entre otros.

Finalmente serán listadas algunas ideas para llevar a cabo, que si bien no son imprescindibles, pueden enriquecer la calidad del producto.

Integración

Implementar interfaces que permitan la integración de la aplicación con herramientas CMS (*Content Management System*), tales como Joomla, WordPress o Drupal por hacer mención a las más populares.

Manejo de Estilos

La aplicación ya maneja el concepto de estilos. Es recomendable el desarrollo de una interfaz de administración de los éstos, para proveer al usuario una forma sencilla de definir estilos personalizados en la hoja de cálculo.

Plantillas

Introducir el concepto de plantillas (*templates*), de forma que la aplicación pueda ser capaz de crear y cargar dicha plantilla en lugar de un documento en blanco.

Macros

Incorporar macros (código embebido), que serían utilizadas para grabar y reproducir acciones elaboradas por el usuario.

Optimización

La herramienta ExtJs es utilizada básicamente para las barras de herramientas y los cuadros de diálogo. Más del 85% del tamaño del código descargado por el navegador proviene de dicha herramienta, y se utiliza un conjunto reducido de sus funcionalidades. Si se busca velocidad en la carga inicial de la aplicación es posible prescindir de ExtJs si se implementan solamente los componentes utilizados.

9 GLOSARIO

1. AJAX: Acrónimo de “Asynchronous Javascript And XML” Es una combinación de tecnologías para el desarrollo de páginas Web interactivas.
2. API: Acrónimo de “Application Programming Interface”, es un conjunto de funciones o procedimientos ofrecidos por una aplicación a modo de interfaz de comunicación.
3. BIFF: Acrónimo de “Binary Interchange File Format”, formato binario de almacenamiento de información utilizado por *Microsoft Excel 2003* y versiones anteriores.
4. Browser: Navegador de Internet.
5. Bugs: Un defecto de software (*computer bug* en inglés).
6. CMS: Sistema de Gestión de Contenidos (Content Management System), aplicación de software que permite la creación y edición de contenidos (típicamente Web) separando el diseño del contenido.
7. Código Abierto: Software desarrollado y distribuido libremente.
8. CSS: Siglas de “Cascading Style Sheets”, es un lenguaje utilizado para estructurar la presentación de páginas Web.
9. CSV: Siglas de “*Comma Separated Values*”, formato de representación de datos separados por comas.
10. DBMS: Software específico que oficia de interfaz entre la base de datos y las aplicaciones que hacen uso de ella.
11. DOM: Acrónimo de “*Document Object Model*”, es una convención para la representación e interacción con objetos en HTML, XHTML y XML.
12. Drupal: sistema de gestión de contenido basado en PHP con licenciamiento GPL desarrollado por Dries Buytaert.
13. Ecma International: Organización para la generación de estándares de sistemas de información y comunicación.
14. ExtJS: Biblioteca de Javascript para aplicaciones Web interactivas.
15. Feng Office: Nuevo nombre de la *Suite* Ofimática *OpenGoo*. (Ver *OpenGoo*).
16. GelSheet: Aplicación Web de manejo de hojas de cálculo realizada como Proyecto de Grado.
17. GNU: Acrónimo recursivo de “*GNU is not Unix*”, es un sistema operativo compuesto enteramente por software libre.
18. GPL II: versión II del licenciamiento GPL (Ver GPL).

19. GPL: Acrónimo de "*General Public License*" es una licencia creada para proteger la libre modificación, distribución y uso de software.
20. GUI: Interfaz gráfica de Usuario, acrónimo de *"Graphic User Interface"*.
21. Hoja de Cálculo: Programa que permite manipular datos numéricos y alfanuméricos.
22. HTML: Siglas de "*HyperText Markup Language*", es un lenguaje para la construcción de páginas Web.
23. HTTP: Siglas de "*HyperText Transfer Protocol"*, es un protocolo de transferencia de texto utilizado en Internet.
24. InnoDB: Tecnología de almacenamiento de datos de código abierto para la base de datos MySQL.
25. Javascript: Lenguaje de programación utilizado principalmente a nivel del Navegador de Internet para la mejora de interfaces y dinámica de las páginas Web.
26. Joomla: sistema de gestión de contenidos Web basado en PHP y MySQL con licenciamiento GPL (Ver CMS).
27. JSON: Acrónimo de "*Javascript Object Notation*", es un formato o notación de intercambio de datos.
28. LGPL: Acrónimo de "*Left General Public License*", es una licencia similar a GPL, con la diferencia principal que puede los programas con esta licencia pueden ser utilizados por programas no GPL.
29. Libro de Cálculo: Conjunto de Hojas de Cálculo.
30. Microsoft Internet Explorer: Navegador Web producto de la empresa Microsoft.
31. Model-View Controller: Es un patrón de arquitectura de software que separa los datos de una aplicación, la interfaz de usuario, y la lógica de control en tres componentes distintos.
32. Mozilla Firefox: Navegador Web producto de la organización Mozilla Foundation.
33. MySQL: Gestor de bases de datos relacional desarrollado por *Sun Microsystems*.
34. ODS-PHP: Conjunto de clases en PHP para la lectura y escritura de archivos ODS.
35. Office Open XML: Formato de archivo usado para representar hojas de cálculo, diagramas, presentaciones y documentos de texto.
36. Ofimática: Equipamiento de Software y Hardware para la manipulación de la información.
37. OLE: Acrónimo de "*Object Linking and Embedding*", es una tecnología de *Microsoft* que permite el enlazado de objetos y documentos.
38. OOXML: Ver Office Open XML.

39. Open Source: Ver Código abierto.
40. OpenGoo: *Suite* Ofimática Web, elaborada en un proyecto de grado de la Facultad de Ingeniería de la Universidad de la República (UDELAR). Predecesora del actual producto Feng Office.
41. ORM: Técnica de programación para convertir datos entre el objeto utilizado y la tabla correspondiente de la base de datos relacional.
42. PDF: Siglas de "*Portable Document Format*", es un formato de documento de texto desarrollado por la empresa Acrobat, adquirido por Adobe.
43. PEAR: Acrónimo de "*PHP Extension and Application Repository*", es un entorno de desarrollo para componentes de código PHP.
44. PHP: Lenguaje de Programación diseñado para el desarrollo de sitios o páginas Web dinámicas.
45. PHPExcel: Conjunto de clases en PHP para la lectura y escritura de diferentes formatos, principalmente XLSX.
46. Plugin: Aplicación que interactúa con una aplicación huésped (por ejemplo un navegador de Internet) para proveer una funcionalidad específica.
47. RSS: Es una familia de formatos codificados en XML. Se utiliza para suministrar a información actualizada frecuentemente.
48. Scrollbar: Barra de desplazamiento en inglés.
49. Sheet: Ver Hoja de Cálculo.
50. Software Libre: Software que puede ser usado, modificado, estudiado y distribuido por sus usuarios.
51. *Spreadsheet*: Ver hoja de cálculo.
52. *SQL Injection*: Técnicas de ataques informáticos que refieren a la inserción de código SQL en variables de entrada no controladas en una aplicación.
53. Stand-Alone: Software que no forma parte, o que no depende de otra aplicación.
54. *Suite* Ofimática: Recopilación de programas para la administración de archivos y documentos.
55. Template: plantilla en inglés.
56. TSV: Siglas de "*Tab Separated Values*", formato de representación de datos tabulados.
57. VBA: Acrónimo de "*Visual Basic for Applications*" es un lenguaje de macros desarrollado por Microsoft para la ampliación de funcionalidades de Microsoft Office

58. Web 2.0: Término que designa a las aplicaciones Web que facilitan la interoperabilidad, intercambio de información, diseño orientado a la usabilidad y colaboración.

59. Web Suite: Suite Ofimática accesible a través de un navegador de Internet.

60. World Wide Web Consortium (W3C): Consorcio internacional abocado a la realización de recomendaciones de estándares para la *World Wide Web*.

61. WordPress: Sistema de Gestión de Contenido (CMS) Web, basado en PHP y MySQL originalmente diseñado para gestionar Blogs.

62. Worksheet: Ver Libro de Cálculo.

63. XHML: acrónimo en inglés de eXtensible Hyper**t**ext Markup Language (lenguaje extensible de marcado de hipertexto), nuevo lenguaje de marcas basado en XML que pretende sustituir el HTML clásico, elaborado por la W3C *(World Wide Web Consortium)*.

64. XLS: Extensión de archivo de las hoja de cálculo del formato BIFF5 de Microsoft.

65. XLSX: Extensión de los archivos de las hojas de cálculo del formato Office Open XML.

66. XML: Siglas de “eXtensible Markup Language”, es un metalenguaje de etiquetas para la definición de gramáticas específicas.

67. XSS: Técnicas de ataque que permiten ejecutar código de "*scripting*", como VBScript o JavaScript, en el contexto de otro sitio web.

68. ZIP: Formato muy utilizado para la compresión de datos como imágenes, programas o documentos.

10 BIBLIOGRAFÍA

1. **Free Software Foundation, Inc.** GNU General Public License, version 2. *GNU Project.* [En línea] 1991. [Citado el: 16 de 1 de 2010.] http://www.gnu.org/licenses/old-licenses/gpl-2.0.html.

2. **Shuen, Amy.** *Web 2.0: A Strategy Guide. Business thinking and strategies behind sucessful Web 2.0 implementationts.* s.l. : O'Reilly, 2008. 0596529961.

3. Opengoo. *Sitio Oficial Opengoo.* [En línea] 1 de 5 de 2009. http://www.opengoo.org.

4. **Power, Daniel.** A Brief History of Spreadsheets. [En línea] Abril de 2009. http://dssresources.com/history/sshistory.html.

5. The XMLHttpRequest Object. *W3C.* [En línea] Abril de 2009. http://www.w3.org/TR/2008/WD-XMLHttpRequest-20080415/.

6. **Aune, Sean P.** Forget Excel: 14 Online Spreadsheet Applications. *Mashable.* [En línea] Abril de 2009. http://mashable.com/2008/02/06/forget-excel-14-online-spreadsheet-applications/.

7. **McLeish, Sheri.** Forrester Research. *http://www.forrester.com/.* [En línea] 18 de Mayo de 2009. http://blogs.forrester.com/adm/09-05-18-microsoft_office_still_owns_desktop_future_staroffice_unclear.

8. **Microsoft.** Nuevo interface de usuario en Office 2007 . *Microsoft Office.* [En línea] 28 de Noviembre de 2010. http://www.microsoft.com/spain/office/eventosonline/trucos33.mspx.

9. CanvasGraph. *Graphing in Javascript.* [En línea] Abril de 2009. http://www.liquidx.net/canvasgraphjs/.

10. JpGraph. *PHP Graph Creating Library.* [En línea] Abril de 2009. http://www.aditus.nu/jpgraph/.

11. PlotKit. *Javascript Chart Plotting.* [En línea] Abril de 2009. http://www.liquidx.net/plotkit/.

12. **Mandel, Theo.** *The Elements of User Interface Design.* s.l. : Wiley , 1997. ISBN 0471162671.

13. **Sun Microsystems.** Distributed Aplication Architecture. [aut. libro] George Reese. *Database Programming with JDBC and Java, Second Edition.* 2006.

14. minify. *Google Code.* [En línea] Setiembre de 2009. http://code.google.com/p/minify/.

15. **Fowler, Martin.** *Patterns of Enterprise Application Architecture.* s.l. : Addison-Wesley, 2002. ISBN 0321127420.

16. Front Controller - Web Application Component Toolkit. [En línea] Junio de 2009. http://www.phpwact.org/pattern/front_controller.

17. What is PHP? *The PHP Group.* [En línea] Abril de 2009. http://www.php.net/manual/en/intro-whatis.php.

18. Home. *MySql.* [En línea] Abril de 2009. http://www.mysql.com/.

19. Introducing JSON. *JSON.* [En línea] Abril de 2009. http://www.json.org/.

20. Products. *Ext JS Client-side Javascript Framework.* [En línea] Abril de 2009. http://www.extjs.com/products/js/.

21. **Deitel, Harvey y Deitel, Paul.** *C++ How to Program.* s.l. : Prentice Hall, 2005. ISBN 0131857576.

22. Licencia GPL Versión 3. *GPL.* [En línea] Abril de 2009. http://www.gnu.org/copyleft/gpl.html.

23. Home Page. *Cascading Style Sheets.* [En línea] Abril de 2009. http://www.w3.org/Style/CSS/.

24. [En línea] http://www.exforsys.com/tutorials/javascript/javascript-object-oriented-features.html.

25. JavaScript Object Notation for Ajax Web services. *SearchSoa.* [En línea] Junio de 2009. http://searchsoa.techtarget.com/tip/0,289483,sid26_gci1224902_mem1,00.html?ShortReg=1&mboxConv=searchSOA_RegActivate_Submit&.

26. HTML 5. *W3C.* [En línea] Agosto de 2009. http://www.w3.org/TR/2009/WD-html5-20090825/.

27. Market Share. *MySql.* [En línea] Agosto de 2009. http://www.mysql.com/why-mysql/marketshare/.

28. **Bray, Tim.** JSON and XML. *Ongoing.* [En línea] Junio de 2009. http://www.tbray.org/ongoing/When/200x/2006/12/21/JSON.

29. ods-php. *SourceForge.* [En línea] Junio de 2009. http://ods-php.sourceforge.net/.

30. Excel specifications and limits. *Microsoft Office Online.* [En línea] Junio de 2009. http://office.microsoft.com/en-us/excel/HP051992911033.aspx?pid=CH062527721033.

31. Calc Guide. *OpenOffice.org Wiki.* [En línea] Junio de 2009. http://wiki.services.openoffice.org/wiki/Documentation/OOoAuthors_User_Manual/Calc_Guide/Introducing_Calc.

32. Features. *Zoho Sheet.* [En línea] Junio de 2009. http://sheet.zoho.com/features.

33. Getting to Know Google Docs. *Google Docs Help.* [En línea] Junio de 2009. http://docs.google.com/support/bin/topic.py?topic=15152.

34. Editgrid vs Google. *EditGrid.* [En línea] Junio de 2009. http://www.editgrid.com/tnc/pkchan/EditGrid_v._Google.

35. **BSC SIGIST.** Standard for Software Component Testing. *Testing Standards Working Party.* [En línea] Junio de 2009. http://www.testingstandards.co.uk/Component%20Testing.pdf.

36. Documents. *PHPExcel.* [En línea] Junio de 2009. http://phpexcel.codeplex.com/wikipage?title=Documents.

37. **Feiler, Jesse.** *Web 2.0 Mashups.* s.l. : McGraw-Hill Osborne, 2007. ISBN 0071496270.

38. **Asleson, Ryan.** *Foundations of Ajax.* s.l. : Apress, 2005. ISBN 1590595823.

39. **Paul J. Deitel, Harvey M. Deitel.** *AJAX, Rich Internet Applications, and Web Development for Programmers.* s.l. : Prentice Hall, 2008. ISBN 0131587382.