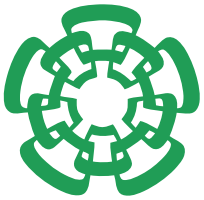


Centro de Investigación y de Estudios Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

Integración de los espacios de comunicación, producción y coordinación en un entorno cooperativo

Tesis que presenta

Ing. Laura Elizabeth Granados Hernández

para obtener el Grado de

Maestra en Ciencias en Computación

Director de la Tesis:

Dra. Sonia Guadalupe Mendoza Chapa

México, Distrito Federal **Diciembre, 2012**

Resumen

Los avances en las tecnologías de la información y de las comunicaciones han cambiado la forma de trabajar de las personas, sobre todo en lo referente a la realización de tareas en forma conjunta. Hoy en día, contamos con sistemas que permiten la interacción entre diferentes individuos, sin importar el tiempo o la ubicación geográfica. El nuevo desafío es desarrollar sistemas que aporten a los colaboradores la percepción de que el trabajo se está realizando en equipo y que, además, faciliten las actividades realizadas por dichos colaboradores.

Para el diseño del entorno cooperativo propuesto se tomó en consideración los aspectos más relevantes de los sistemas *Groupware*, tales como: memoria grupal, roles de usuario, protocolos de colaboración y percepción grupal, con el objetivo de integrarlos en tres espacios de trabajo: *producción*, *comunicación* y *coordinación*. Dichos espacios están enfocados en aumentar la eficacia de los colaboradores.

El entorno cooperativo propuesto tiene como principal objetivo facilitar la edición conjunta, donde cada colaborador puede tener asignado un rol diferente de acceso a la información (documento o imagen). En comparación con los entornos cooperativos actuales se buscó: 1) ofrecer a los colaboradores la posibilidad de que la edición de documentos o imágenes se pueda hacer de forma síncrona o asíncrona; 2) facilitar la comunicación de los colaboradores con un proceso de *deixis* o señalamiento, desde el espacio de comunicación hacia el espacio de producción; 3) manejar la coordinación de actividades, mediante roles de acceso a la información y la utilización de candados; 4) aprobar el contenido del documento o la imagen por un comité integrado por uno o varios colaboradores; y 5) adaptar el sistema de acuerdo a las diferentes circunstancias de una organización conforme al contexto de uso de dicho sistema.

Palabras clave: *Groupware, Trabajo Cooperativo Asistido por Computadora, editores grupales, editores cooperativos en tiempo real, aplicaciones conscientes de contexto*

Abstract

Advances in information technology and communications have changed the way in which people work, especially regarding collaborative work. Nowadays, there are systems that allow interaction among individuals, regardless the geographical space and time. The new challenge is to develop systems that provide collaborators with the perception that the task is being performed by a team and that also facilitate the collaborators' activities. The design of the proposed cooperative environment takes into account the most relevant aspects of *Groupware* systems, such as group memory, user roles, collaboration protocols and group perception, in order to integrate them into three functional spaces: *production*, *communication* and *coordination*, all focused on increasing the efficiency of employees. The main objective of the proposed cooperative environment is to facilitate collaborative editing of shared documents where each collaborator has a different role to access information (document or image). In comparison with current cooperative environments, we search for: 1) allowing collaborators to write documents and images in synchronous and asynchronous ways; 2) facilitating communication among employees by means of a *deixis* or signaling process from the communication space to the production space; 3) managing the coordination of activities by means of access roles and locks; 4) approving the document or image contents by means of a committee consisting of one or more collaborators; 5) adapting the system to the different situations of an organization, according to the context of use of this system.

Keywords: *Groupware, Computer-Supported Cooperative Work (CSCW), group editors, real-time cooperative editors, context-aware applications*

Dedicatoria

A:

Dios,
por darme la oportunidad de vivir y estar conmigo en cada paso que doy, por fortalecer mi corazón e iluminar mi mente y por haber puesto en mi camino a aquellas personas que han sido mi soporte y compañia durante todo el periodo de estudio.

Mis padres,
por ser el pilar fundamental en todo lo que soy, en toda mi educación, tanto académica, como de la vida, por el incondicional apoyo que me proporcionaron.

Agradecimientos

A:

Dra. Sonia Mendoza,

por haberme apoyado en todo momento, por sus compartir conmigo sus conocimientos y consejos, por la asesoría brinda durante la realización de esta tesis de maestría; por soportar mis malos ratos de estudiante y la motivación constante para finalizar este grado de estudios.
¡Gracias!

CINVESTAV,

por haberme brindado la oportunidad de pertenecer a su extraordinaria comunidad y estudiar un posgrado de la más alta calidad.

CONACyT,

por el apoyo económico proporcionado durante el periodo de estudios de esta tesis de maestría, sin el cual no habría podido realizarse.

Mis hermanos (Estela y Ernesto) y mi tío Marcial,

por ser un gran ejemplo a seguir, por siempre preocuparse por mi bienestar y superación, por todo el cariño, apoyo, compresión y los consejos que me brindaron en todo momento.
¡Gracias a ustedes!

Herman Peralta,

porque desde el primer momento que lo conocí se ha convertido en una inspiración y un ejemplo a seguir, por todo su apoyo y comprensión en los malos ratos y por todos los bellos momentos que hemos compartido.
¡Gracias amor!

Dr. José G. Rodriguez,

por los consejos y la motivación que me brindó cada vez que toque su puerta en busca de ayuda, por el tiempo invertido en la revisión de esta tesis de maestría

Dra. Ma. Lizbeth Gallardo,

por el tiempo invertido en la revisión de esta tesis de maestría y sus valiosos consejos.

Dr. Carlos Coello,

por el consejo que me brindó y darme el impulso necesario para alcanzar el grado de estudios.

Christian Cruz,

por su valiosa amistad, por todo su apoyo y la motivación que me ha brindado y por sus valiosos consejos que me sirvieron para lograr este grado de estudios. ¡Gracias!

Cinvescuates,

por todo el apoyo mutuo que nos brindamos en nuestra formación profesional y su valiosa amistad.

Profesores del Departamento de Computación,

por todos los conocimientos compartidos tanto dentro como fuera del salón de clases.

Sofi,

por todo el apoyo, los consejos y servicios que me ha brindado durante mi estancia en el CINVESTAV.

Amigos y familiares,

por su amistad y cariño y su participación directa o indirecta en la elaboración de esta tesis.

Lucky, mi chachorro,

por ser mi dulce compañía y por brindarme momentos de distracción todos los días.

Índice general

Índice de figuras

Índice de tablas

Capítulo 1
Introducción

Este capítulo presenta un panorama general del proyecto de tesis de maestría. Primeramente, se expone el contexto de investigación, los antecedentes y la motivación que dió origen al diseño y a la implementación del Entorno Cooperativo Consciente de Contexto propuesto. El capítulo se dividió en cinco secciones: la sección 1.1 presenta el contexto de investigación de la presente tesis de maestría, la sección 1.2 presenta los antecedentes, la sección 1.3 presenta el planteamiento del problema a resolver, la sección 1.4: presenta los objetivos del proyecto de tesis de maestría y finalmente, la sección 1.5: presenta la organización de los siguientes capítulos del presente documento de tesis.

1.1. Contexto de investigación

El Trabajo Cooperativo Asistido por Computadora (*Computer Supported Cooperative Work* o *CSCW* por sus siglas en inglés) es un área de investigación interdisciplinaria de las ciencias de la computación y las ciencias sociales [Decouchant et al., 2009]. Esta disciplina estudia las características que deben tener los sistemas computacionales para dar soporte al trabajo producido por un grupo de personas en los diferentes escenarios de colaboración dentro de una organización.

En particular, el área de investigación de *Groupware* se encarga de diseñar las diferentes tecnologías (software y hardware) que den soporte y faciliten a un grupo de trabajo alcanzar los objetivos o las metas que tienen en común, mediante una interfaz del entorno compartido [Ellis et al., 1991].

Para poder diseñar un sistema *Groupware* se deben considerar las contribuciones de otras seis ramas de la ciencia de computación, con las cuales *Groupware* mantiene una relación muy estrecha. La Figura 1.1 muestra un esquema que relaciona las principales áreas

de investigación o disciplinas que se encuentran directamente relacionadas con *Groupware*.

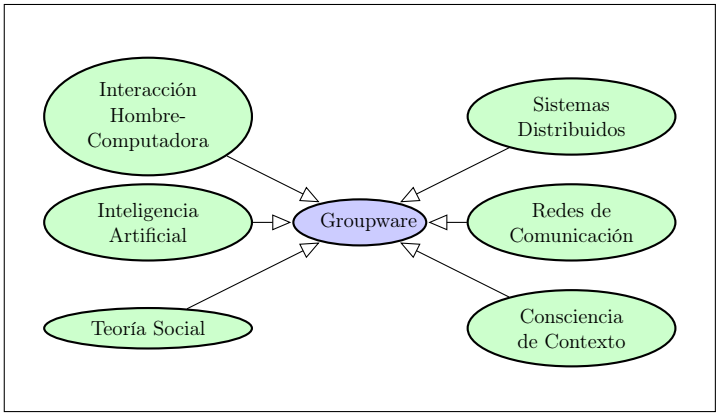


Figura 1.1: Disciplinas relacionadas con el diseño de los sistemas *Groupware*

Los sistemas *Groupware* no se concentran en una sola disciplina, sino que adoptan y combinan los avances y descubrimientos de otras disciplinas (véase Figura 1.1) para ofrecer servicios y herramientas que faciliten el trabajo cooperativo de un grupo de personas. Dichos servicios y herramientas toman en consideración las necesidades que presenta el grupo de trabajo al interactuar por medio de una o más computadoras [Ellis et al., 1991].

A continuación se muestra la influencia de cada una de estas disciplinas.

La **Teoría Social** ayuda a entender y describir los procesos que están presentes en la realización de una tarea de forma grupal. Al conocer las características de cada proceso, se pueden diseñar servicios y herramientas que contribuyan en la realización de la tarea, facilitando el trabajo de los colaboradores, en cuanto a la coordinación de actividades y la toma de decisiones.

La **Inteligencia Artificial** transforma a la computadora de un agente pasivo a un agente activo que facilita la colaboración entre los miembros de un grupo de trabajo, estableciendo diferentes protocolos para la planeación de actividades, la colaboración y la negociación entre dichos miembros.

Las personas interactúan con una computadora a través de una interfaz de usuario, cuyo diseño se estudia en la disciplina de la **Interacción Hombre-Computadora**. La interfaz de usuario puede ser considerada como uno de los componente más importantes del sistema, ya que las personas sólo conocen al sistema a través de su interfaz y los servicios que ésta puede ofrecerle [Saltiveri, 2005]. Como se mencionó anteriormente, los sistemas *Groupware* utilizan una interfaz de el entorno compartido, donde se requieren componentes que permitan la realización de las tareas compartidas y que proporcionen

una percepción grupal[1] de las actividades realizadas por los colaboradores que interactúan por medio del sistema.

Los sistemas *Groupware* utilizan una red de computadoras para comunicarse e intercambiar información, utilizando recursos propios o ajenos, lo cual agiliza la realización de las actividades asignadas a cada participante en la tarea común. La disciplina de **Redes de Comunicación** hace referencia al conjunto de elementos y procedimientos que permiten mantener este intercambio de información a distancia.

Los **Sistemas Distribuidos** permiten ver un sistema geográficamente distribuido como una única entidad, donde los componentes del sistema se localizan en computadoras conectadas a una red de comunicación; la coordinación de actividades de las computadoras se realiza mediante el intercambio de mensajes. Coulouris et al. describen las características que deben tener los sistemas distribuidos, con el fin de mantener la consistencia del estado global de los recursos compartidos. Dichas características son: compartir recursos, apertura, concurrencia, escalabilidad, tolerancia a fallos y transparencia [Coulouris and Dollimore, 1988]; estás características que son de gran relevancia para los sistemas *Groupware*, ya que permiten tener una consciencia grupal de la tarea que se está realizando de forma conjunta.

El término de conciencia de grupo es definido por Dourish and Bly como "un conocimiento de las actividades de otros que provee un contexto para las actividades propias" [Dourish and Bly, 1992].

Una disciplina no estudiada por Ellis pero que podemos considerar de gran relevancia en la búsqueda de una consciencia grupal es la **Consciencia de Contexto**. Esta disciplina recopila información contextual de los aspectos del mundo físico para conocer las circunstancias que son relevantes para el usuario o el sistema y así, adaptarse a los cambios; su propósito es ofrecer servicios de acuerdo a la circunstancias que se presente. La Consciencia de Contexto puede utilizarse para proporcionar a los participantes información complementaria que es de gran utilidad en la formación de una consciencia grupal, al tomar en consideración aspectos como el tiempo, la identidad y ubicación, para establecer las características presentes en un momento determinado y ofrecer servicios que faciliten la interacción de los colaboradores del sistema de forma automática.

Gracias a las contribuciones que tienen estas seis disciplinas en los sistemas *Groupware*, se puede ver a *Groupware* como el conjunto de métodos, herramientas y metodologías que buscan facilitar la realización de una tarea por un grupo de personas , aumentando la eficacia de cada colaborador en cuanto a tres aspectos importantes (desde el punto de

[1]La percepción es el proceso cognoscitivo que permite interpretar y comprender un entorno. El sistema debe proporcionar tanto la información sobre los miembros del grupo (e.g. quiénes están conectados y qué hacen éstos), así como información referente a los cambios efectuados sobre los elementos de la tarea, con el fin de proporcionar un conocimiento compartido.

vista de los usuarios del sistema): la *comunicación* de los usuarios, la *coordinación* de las actividades y la *cooperación* intencionada de los miembros del grupo de trabajo.

Como se mencionó anteriormente, las personas que utilizan un sistema *Groupware* pueden estar distribuidos en espacio (geográficamente) y/o en tiempo. De ahí se desprende la taxonomía espacio-temporal de los sistemas *Groupware*, en la cual se puede agrupar a los sistemas como: *sistemas en tiempo compartido* donde la interacción es de forma asíncrona y *sistemas en tiempo real* donde la interacción es al mismo tiempo.

La línea que divide a los sistemas *Groupware* de aquellos que no lo son es muy grande: si el sistema incluye un entorno compartido entonces se debe considerar como parte de *Groupware*. A continuación, se muestra una taxonomía en cuanto a las características o funcionalidades que pueden ser consideradas para clasificar a un sistema dentro de *Groupware* según el nivel de aplicación:

1. **Sistemas de mensajería:** intercambio de mensajes entre un grupo de personas, e.g., correo electrónico, conferencias (inteligentes), etc.
2. **Editores multi-usuario:** herramientas que permiten editar un mismo documento a un grupo de personas.
3. **Sistema de soporte a decisiones grupales y salas de reunión:** facilitan la exploración de problemas no estructurados en las actividades del grupo y mejoran la toma de decisiones. Una de sus implementaciones son las salas de reunión electrónicas.
4. **Conferencias electrónicas:** establecen un medio de comunicación a través de una computadora. Se pueden clasificar en tres tipos:
   - **Conferencia electrónica en tiempo real**: las personas interactúan a través de sus computadoras por medio de textos enviados de forma sincronizada.
   - **Teleconferencia**: las personas interactúan a través de sus computadoras por medio de video-llamadas.
   - **Conferencia de escritorio**: se comparte el escritorio de una computadora a través del envío del mismo como video.

En estas aplicaciones se puede observar que: 1) los usuarios encargados de la realización de tareas específicas se muestran como *agentes inteligentes*, 2) las acciones que realiza cada colaborador en los elementos del sistema manejan un proceso de *coordinación*, 3) se tiene un sistema que da *soporte a la toma de decisiones* y 4) para el intercambio de ideas, éste tipo de aplicaciones, ofrecen un proceso de *comunicación*.

En la mayoría de los sistemas *Groupware* existe un traslape en estas funcionalidades, con el fin de ofrecer más y mejores servicios a los usuarios del sistema. Se busca que las funcionalidades satisfagan las necesidades que presenta el grupo de trabajo cuando interactúan

por medio de computadoras al realizar una tarea específica.

La presente tesis de maestría toma en consideración la evolución de los editores multi-usuario, también conocidos como editores cooperativos, para el estudio de sus características y deficiencias, con el objetivo de diseñar un entorno cooperativo consciente de contexto que permita la edición de documentos.

1.2. Antecedentes

A pesar de que la investigación de CSCW comienza en la década de los 80's, la idea de un sistema cooperativo nació años antes, Engelbart en el artículo "*The mother of all demos*[1]" explica las principales características que debe tener un sistema computacional para permitir a un grupo de personas la edición de un documento de forma conjunta, su propuesta se conoce como el primer editor multi-usuario [Engelbart, 1968].

La siguiente versión de los editores multi-usuarios fueron los **repositorios de versiones**, también conocidos como *Subversion*. Este tipo de aplicaciones permite la edición de documentos digitales[2], la comparación de los cambios realizados por cada colaborador de forma aislada, mediante la generación de múltiples copias, y el seguimiento de los cambios, mediante un historial de versiones [Fraser, 2011].

Los sistemas de actualización de versiones que ofrecen los sistemas operativos actuales (Mac OS, Microsoft Windows y Linux) son un claro ejemplo del funcionamiento de la gestión del control de versiones. Estos sistemas proporcionan un servicio de actualización de archivos (documentos) a varios clientes, los clientes al conectarse a Internet, examinan el servidor del sistema operativo que contiene las actualizaciones de los componentes del sistema en busca de alguna actualización disponible para ser descargada e instalada automáticamente en la computadora[3].

La evolución de los editores multi-usuario continúa hasta lo que hoy conocemos como **editores cooperativos síncronos**. Estas aplicaciones permiten a un grupo de personas editar el mismo documento desde diferentes dispositivos electrónicos y al mismo tiempo. Los editores cooperativos toman en consideración las tendencias actuales de los editores para que el usuario final se sienta familiarizado con la interfaz, lo que le permite a los editores cooperativos ofrecer servicios y/o herramientas que faciliten el trabajo en equipo,

[1] Engelbart establece los principios de los editores cooperativos mediante los resultados obtenidos de un experimento que realizó en un laboratorio de Stanford que contaba con 12 computadoras, conectadas en red, mediante las cuales se podía compartir información e interactuar al mismo tiempo.

[2] No se permite realizar la edición de un documento al mismo tiempo por dos o más usuarios.

[3] Los sistemas de actualización de versiones se basan en una arquitectura cliente-servidor para coordinar las actualizaciones.

lo cual motive la utilización del sistema.

Whalen menciona que en la actualidad existen dos técnicas para compartir la información en editores cooperativos [Whalen, 1997]. Algunos editores cooperativos utilizan la técnica de replicación para compartir el documento, lo cual genera múltiples copias del mismo, cuya actualización se realiza a petición del usuario. Otros editores cooperativos mantienen solo una versión del documento, la cual es compartida a todos los colaboradores a la vez; en este tipo de sistemas la edición de la información se realiza mediante la segmentación del documentos y el acceso a cada segmento es por turnos, los colaboradores ven reflejados los cambios realizados por los otros miembros del equipo de forma automática.

Estas aplicaciones requieren tener un control de concurrencia. Los editores cooperativos que utilizan la técnica de replicación usan candados en la edición de documentos o bien comparan las versiones de cada usuario y notifican las inconsistencias presentes en los datos. Las aplicaciones que mantienen una sola versión no manejan un control de concurrencia pero incluye un proceso de notificación de cambios. A cada colaborador se le asigna un color, con el cual se resalta en la interfaz de los demás colaboradores el texto ingresado, por lo que el no contar con un control de concurrencia provoca problemas de coherencia en la información.

Cuando se requiere discutir ideas entre los colaboradores, los editores cooperativos ofrecen la utilización de notas o comentarios dentro de un documento o en algunos casos se llega a incluir una sala de conversación con el fin de agilizar la interacción.

Aunque existe una amplia gama de propuestas (académicas y comerciales), que aportan características importantes a la edición cooperativa, pocas han sido de gran utilidad para la sociedad actual. Esto se debe al amplio campo de aplicación que pueden tener estos sistemas (documentación técnica, científica, médica, etc.) y que en ocasiones existen ciertas particularidades requeridas para dar un buen soporte al equipo de trabajo, lo que limitaba su generalización para uso en otros campos.

Internet ha favorecido el desarrollo y uso de los editores cooperativos, los más conocidos son las *wikis* y *Google Docs*, los cuales retoman una gran parte de la investigación sobre la edición cooperativa realizada durante 20 años.

En 1995 surgió el concepto de *wiki* como un sistema de gestión de contenido colaborativo que permite a los usuarios crear y editar libremente el contenido de páginas web, el ejemplo más conocido es Wikipedia.

Años más tarde la compañía Google ofreció una herramienta de edición cooperativa en tiempo real llamada *Google Docs*. Google Docs permite compartir documentos con varios usuarios, facilitando la visualización y edición de éstos en tiempo real. Aunque no cuenta con un proceso de notificación de cambios, ofrece una herramienta comunicación

vía correo electrónico, mensaje instantáneo, llamada o videoconferencia lo cual facilita la interacción de los colaboradores pero no existe una interacción entre dichas herramientas.

1.3. Planteamiento del problema

1.3.1. Definición

Para que los sistemas *Groupware* cumplan con el objetivo de facilitar a un grupo de trabajo la realización de una tarea, requieren tener una interfaz del entorno compartido que incluya: 1) un *espacio de producción*, que permita realizar la tarea en conjunto, 2) un *espacio de coordinación*, que administre las actividades (sobre los elementos de la tarea) realizadas por cada participante, así como proveer un mecanismo para la toma de decisiones y 3) un *espacio de comunicación*, para el intercambio de ideas.

Una de las principales motivaciones para realizar el presente proyecto de tesis de maestría, es que en la actualidad no existe un entorno cooperativo que integre los tres espacios propuestos: *producción*, *comunicación* y *coordinación*. Como se pudo observar en los ejemplos antes mencionados, algunas aplicaciones sólo incluyen el espacio de producción y coordinación, mientras que otras incluyen el espacio de producción y comunicación.

La investigación realizada como parte de la presente tesis de maestría se basa en una tarea de uso general para cualquier organización: la edición de documentos. Existen diferentes tipos de editores de documentos, cuya funcionalidad depende del tipo de información que almacena (texto, imagen, sonido o imagen en movimiento). Para el caso de esta investigación, decidimos enfocarnos en *la edición de documentos de texto e imágenes vectoriales*, al ser ésta una actividad cotidiana.

Erkens et al. clasifica las actividades que suelen seguirse en la escritura de un documento en tres: 1) la planificación (creación, organización y coherencia de contenido), 2) la traducción de las ideas en palabras y 3) la revisión del texto [Erkens et al., 2002].

El proceso de revisión en ocasiones implica la participación de otras personas para asegurar que las ideas escritas mantienen una coherencia y orden, así como aportar ideas que enriquezcan el contenido del documento. La participación conjunta e intencionada en la edición y revisión de un documento convierte a éste en un proceso colectivo.

1.3.2. Problemática en los entornos cooperativos existentes

Los retos a los que se enfrentan los sistemas cooperativos es proporcionar una percepción grupal del sistema, lo que requiere una buena sincronización en la interacción de las acciones que se realizan sobre los componentes de la tarea conjunta a realizar, sobre todo

cuando los participantes se encuentran geográficamente distribuidos. Particularmente, en los editores cooperativos se presentan los siguientes problemas que merman la productividad de los miembros del equipo de trabajo:

1. **Edición de documentos**: al editar un documento, el usuario debe cambiar constantemente entre un editor de texto y un editor de imágenes debido a que no se cuenta con un entorno que comprenda ambas funciones.

2. **Interacción de los colaboradores**: limita el proceso de comunicación que tienen los colaboradoras al realizar la edición de un documento, ya que solo permiten ingresar notas o comentarios dentro del documento.

3. **Salas de conversación**: ofrecen un mecanismo más rápido para la interacción de los colaboradores, pero no es un medio eficaz para transmitir las ideas, sobre todo cuando se requiere discutir una sección del documento que se está editando. Por lo general, cuando un grupo de trabajo utiliza una sala de conversación para discutir una sección de un documento compartido, se requiere indicar la ubicación exacta en el documento (e.g. capítulo uno, página tres, párrafo dos) de la que se ésta hablando; esto genera confusión en los colaboradores y evita enfocarse en el tema a discutir, tal como se haría en una interacción cara a cara donde sólo se señala la sección y se exponen las ideas respecto.

4. **Acceso a la información**: un documento se puede compartir en un grupo de trabajo mediante invitación, pero cuando un documento es muy extenso o requiere una participación interdisciplinaria, sería recomendable limitar el acceso a las secciones que competen a cada participante.

5. **Estructura jerárquica del documento**: para poder tener un acceso segmentado a la información segmentada, los usuarios se requiere tener una organización jerárquica del documento, con el fin de mantener un orden en la información, tal como se puede organizar un documento en LaTeX.

6. **Roles de usuario**: muy pocos entornos cuenta con roles de usuario para establecer permisos de acceso, edición y aprobación de la información, tal como lo utiliza el editor cooperativo Alliance que cuenta con cuatro tipos de roles de usaurio (nulo, lector, escritor y administrador) [Decouchant et al., 1999].

7. **Toma de decisiones**: no es considerada en los editores cooperativos, pero un proceso que permita decidir si una sección o documento está correcta o aprobada por los autores que tienen el papel de revisor, sería de gran ayuda para el grupo de trabajo.

1.3.3. Principales aportaciones

El presente trabajo de tesis busca proponer mecanismos que den solución a los problemas antes mencionados, mediante el diseño y la implementación de un entorno cooperativo que integre:

I. **Espacio de producción** que permita la edición cooperativa de:
   - documentos de texto HTML
   - imágenes vectoriales

II. **Espacio de coordinación** que facilite la coordinación de actividades de los colaboradores mediante:
   - el uso de candados
   - el uso roles de acceso a la información
   - la generación de una estructura jerárquica de documentos compartidos
   - la toma de decisiones que permita la aprobación de los documentos terminados

III. **Espacio de comunicación** mediante una sala de mensajes instantáneos que incluya un proceso de *deixis* o señalamiento desde el espacio de comunicación hasta el espacio de producción

1.4. Objetivos del proyecto

1.4.1. Objetivo General

El objetivo principal es diseñar e implementar un entorno cooperativo, al que se llamó *Misho*, que integre los espacios de *producción*, *coordinación* y *comunicación*.

El *espacio de producción* comprende la tarea común que realizan los colaboradores del equipo de trabajo; está integrado por un editor de texto y un editor de imágenes vectoriales.

El *espacio de coordinación* establece un orden en el conjunto de las actividades realizadas por los colaboradores que permite obtener un resultado coherente. También, genera un conjunto de reglas que permiten el correcto funcionamiento del entorno cooperativo. El proceso de coordinación utiliza: roles de usuario y permisos de edición mediante candados, la generación de una estructura jerárquica de los documentos y la toma de decisiones que permita la aprobación de documentos terminados.

El *espacio de comunicación* permite el intercambio de ideas mediante un proceso de envío de mensajes eficaz, mediante la utilización de una sala de mensajería instantánea que

integra un proceso de *deixis* sobre el espacio de producción.

1.4.2. Objetivos Particulares

Diseñar e implementar:

1. *Misho*, el entorno cooperativo propuesto, cuya interfaz gráfica que integre un editor de texto, un editor de imágenes vectoriales y un sala de conversación vía mensaje instantáneo en la misma ventana.

2. Un proceso de *deixis*, i.e. un mecanismo de señalamiento desde el espacio de comunicación al espacio de producción, con el cual el usuario pueda señalar secciones del documento o figuras dentro de la imagen. De esta forma se facilita la comunicación entre los colaboradores cuando hacen referencia a los elementos tanto del documento como de la imagen, dado que dichos elementos son visualizados por los colaboradores mediante un mecanismo que los resalta de los demás elementos.

3. Un proceso de comunicación para compartir información entre los diferentes colaboradores.

4. Un proceso de coordinación en la edición del documento o imagen mediante candados, que además establezca los permisos de acceso a la información mediante roles de usuario.

5. El proceso de aprobación de la información.

1.5. Organización del documento

El presente documento de tesis de maestría está estructurado en cinco capítulos (véase Figura 1.2).

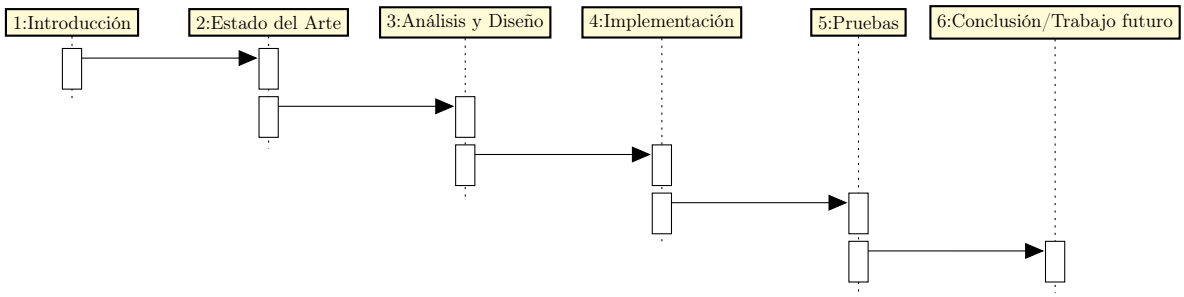


Figura 1.2: Organización del documento

El capítulo 2 describe la investigación realizada para este trabajo de tesis de maestría. Este capítulo consta de cuatro secciones: la sección 2.1 (**conceptos relevantes**) expone los conceptos relacionados con las áreas de investigación de *Trabajo Cooperativo Asistido por Computadora*, *Groupware* y *Sistemas Conscientes de Contexto*; la sección 2.2 (**trabajos relacionados**) describe brevemente el entorno cooperativo propuesto y los trabajos relacionados -entornos cooperativos y editores cooperativos- con el fin de conocer las principales características que ofrecen; la sección 2.3 (**criterios de comparación**) muestra los criterios que se utilizaron para hacer una comparación de las características de los sistemas *Groupware* y finalmente la sección 2.4 (**comparación**) realiza una comparación (bajo los criterios descritos en la sección anterior) de los trabajos relacionados.

Las contribuciones del presente trabajo de tesis de maestría se reflejan en los capítulos 3 y 4. El capítulo 3 presenta los problemas a los que se enfrentan los grupos de trabajo al no tener integrados los tres espacios funcionales de *Groupware* y las técnicas que se estudiaron para dar solución a estos problemas como parte del análisis del problema, además se da a conocer el análisis que se realizó para el diseño de *Misho*. El capítulo 4 muestra la implementación de dicho diseño utilizando en patrón de desarrollo Modelo-Vista-Controlador, describiendo cada uno de los módulos que se diseñaron para cada los espacios de comunicación, producción y coordinación .

El capítulo 5 describe las pruebas realizadas en *Misho* para comprobar su funcionamiento y verificar que se cumpliera el objetivo de facilitar la edición cooperativa de documentos e imágenes a un grupo de trabajo.

Finalmente el capítulo 5 se dan a conocer las conclusiones a las que se llegaron durante la investigación, análisis y desarrollo del *Misho*. Además de incluir un apartado en el que describimos cual puede ser el trabajo futuro que se puede desarrollar sobre *Misho* para facilitar un poco más el trabajo cooperativo.

Capítulo 2
Estado del arte

El capítulo anterior mostró un panorama general del contexto de investigación de la presente tesis de maestría. Sin embargo, es necesario profundizar en los temas relacionados antes de comenzar con el diseño de *Misho* el entorno cooperativo propuesto. Este capítulo consta de cuatro secciones: la sección 2.1 (**conceptos relevantes**) expone los conceptos relacionados con las áreas de investigación de *Trabajo Cooperativo Asistido por Computadora*, *Groupware* y *Sistemas Conscientes de Contexto*; la sección 2.2 (**trabajos relacionados**) describe brevemente el entorno cooperativo propuesto y los trabajos relacionados -entornos cooperativos y editores cooperativos- con el fin de conocer las principales características que ofrecen; la sección 2.3 (**criterios de comparación**) muestra los criterios que se utilizaron para hacer una comparación de las características de los sistemas *Groupware* y finalmente la sección 2.4 (**comparación**) realiza una comparación (bajo los criterios descritos en la sección anterior) de los trabajos relacionados.

2.1. Conceptos relevantes

El trabajo en equipo puede verse como la suma de las aportaciones individuales, pero si no existe una coordinación o comunicación entre los participantes, difícilmente se puede obtener un resultado coherente.

El término **trabajo cooperativo** se usa para describir el trabajo realizado por un grupo de personas que tienen una meta común. Marx en 1867, fue el primero en utilizar este término, definiéndolo como múltiples individuos que trabajan conjuntamente de forma consciente en un mismo proceso de producción o en procesos diferentes pero relacionados entre sí [Mitchell, 1996]. El trabajo cooperativo permite realizar tareas que de forma individual tomarían un tiempo considerable debido a su complejidad. Las personas que forman parte del grupo de trabajo se conocen con el nombre de colaboradores (en adelante se utilizará este término). Es importante conocer cuál es la interacción de cada colabo-

rador con el sistema y con los demás colaboradores durante la realización de la tarea común, con el fin de proporcionar herramientas o mecanismos que faciliten y mejoren el desempeño de los miembros del grupo de trabajo.

Uno de los principales retos que enfrentan los sistemas computacionales que dan soporte a los grupos de trabajo, es el control de los recursos compartidos debido a que pueden ser utilizados en cualquier momento y por cualquier miembro del grupo de trabajo. Si no se tiene un control y administración de los recursos compartidos se puede ocasionar que el resultado obtenido no sea el esperado o tenga problemas de coherencia.

Esto provoca que las actividades de cada colaborador sobre los recursos compartidos tengan una interdependencia y requieran ser administradas por el sistema. El sistema establece el momento en que el recurso compartido puede ser utilizado por cada colaborador para mantener una coherencia en el estado del mismo y poder alcanzar la meta común más fácilmente.

El término **Trabajo Cooperativo Asistido por Computadora**, también conocido por su acrónimo en inglés CSCW (*Computer Supported Cooperative Work*), connota el área de investigación interdisciplinaria de las ciencias sociales y la computación. CSCW se encarga del estudio de las necesidades que presentan los grupos de trabajo, al interactuar por medio de una computadora para proponer soluciones tecnológicas que den soporte a dichos grupos. CSCW se explica más a detalle en la sección 2.1.1.

Groupware es la rama de CSCW encargada de la tecnología utilizada en los sistemas computacionales. El propósito de *Groupware* es coordinar las actividades de los colaboradores de un grupo de trabajo cuando realizan una tarea común. *Groupware* hace referencia al desarrollo de software y hardware que da soporte y facilita el trabajo cooperativo. Se explican los conceptos relacionados a *Groupware* en la sección 2.1.2.

Los sistemas *Groupware* deben tomar en consideración que los colaboradores del equipo de trabajo pueden estar distribuidos en espacio y tiempo, por lo que es recomendable que dichos sistemas sean conscientes del contexto de uso en tres factores principalmente: lugar, identidad y tiempo. En la sección 2.1.5 se explica el área de investigación conocida como **Consciencia de contexto**, la cual estudia las características de los sistemas que consideran el contexto de uso para adaptarse a los cambios de circunstancias de su ambiente físico, ofreciendo servicios o herramientas adecuadas al usuario.

2.1.1. Trabajo Cooperativo Asistido por Computadora - CSCW

En 1984, Greif y Cashman realizaron un taller interdisciplinario, con el fin de conocer las características que requieren los sistemas computacionales para facilitar el trabajo cooperativo a través de una red de computadoras. Greif utilizó por primera vez el término de Trabajo Cooperativo Asistido por Computadora (CSCW) y lo definió como “una vía

para describir cómo la tecnología de las computadoras pueden ayudar a los usuarios a trabajar de forma cooperativa"[Greif, 1988].

Posteriormente Schmidt and Bannon describieron a CSCW como el esfuerzo por comprender la naturaleza y las características del trabajo cooperativo, con el objetivo de diseñar adecuadas tecnologías basadas en computadoras [Schmidt and Bannon, 1992].

El objetivo de CSCW es observar la forma en que las personas interactúan al momento de colaborar entre ellas; con ello, se conocen las necesidades de los colaboradores y se proponen una serie de consideraciones en el desarrollo de herramientas y/o servicios que faciliten la cooperación, utilizando una red de computadoras.

CSCW establece características que deben tomarse en cuanta en el diseño de los sistemas computacionales, considerando los siguientes escenarios:

- Los colaboradores pueden estar situados en diferentes entornos o situaciones de trabajo.
- Los colaboradores puede tener distintas responsabilidades, perspectivas y tendencias.
- Los colaboradores interactúan y se condicionan mutuamente al realizar su trabajo.

2.1.2. Groupware

El término *Groupware* connota una visión estrecha del campo de CSCW, haciendo énfasis al desarrollo de tecnologías de software y hardware que dan soporte y facilitan el trabajo cooperativo de un grupo de personas. En la literatura se utilizan ambos términos (en ocasiones como sinónimos) para definir a los entornos que dan soporte al trabajo cooperativo usando una red de computadoras, aunque algunos autores diferencian a *Groupware* de CSCW porque los primeros hacen referencia a la tecnología utilizada, mientras que los segundos incluyen el estudio de la teoría social del grupo de trabajo.

Ellis et al. definen *Groupware* como un sistema basado en computadoras que da soporte a un grupo de individuos comprometidos en una meta común [Ellis et al., 1991]; el sistema debe proveer una interfaz del ambiente compartido. Por su parte, Jhonson-Lenz define *Groupware* como un sistema basado en computadoras que complementa los procesos de un grupo social [Cantero et al., 2001]. Esta definición fue ampliada posteriormente por Goldberg a un sistema que permite interactuar a las diferentes entidades de un grupo de individuos basándose en una estructura tecnológica [Cantero et al., 2001].

El objetivo de *Groupware* es proveer un conjunto de métodos, medios y herramientas que aumenten la eficacia de los miembros del grupo de trabajo, en cuanto a: **comunicación**,

coordinación y **cooperación** [Ellis and Wainer, 1994], i.e. es importante proveer de un medio de comunicación, un conjunto de métodos para la coordinación de actividades y las herramientas necesarias que faciliten la cooperación de los colaboradores.
A continuación, se presenta una descripción de estos tres conceptos para conocer la importancia que tienen en los sistemas *Groupware*.

La **comunicación** hace referencia al proceso eficaz para el intercambio de mensajes. Busca un proceso de comunicación donde el emisor (el individuo encargado de enviar el mensaje con la información) y el receptor (el individuo que recibe el mensaje) perciban el mismo concepto, esperando gastar el mínimo de recursos. En la figura 2.1, se ejemplifica un proceso de comunicación donde la idea es transmitir un mensaje que permita que tanto el emisor como el receptor entiendan que se está hablando de un árbol, no de otra idea (como podría ser un libro, una manzana, etc); si ambos logran entender el mensaje, la comunicación ha sido satisfactoria. Este proceso es un reto si la interacción se realiza a través de computadoras, debido a que los gestos que podemos utilizar para dar entender las ideas se limitan al envío de palabras o imágenes relacionadas.

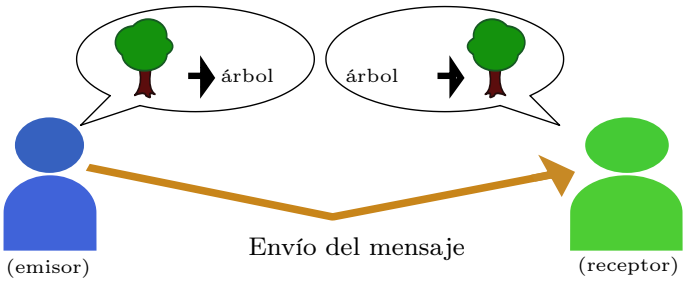


Figura 2.1: Proceso de comunicación eficaz

La **coordinación** es la acción de asegurar que el equipo de trabajo está trabajando conjuntamente para alcanzar una meta común. La coordinación analiza las dependencias existentes entre los colaboradores y sus acciones, al realizar la tarea común y procura dar un orden a las actividades realizadas por cada colaborador, generando un enlace coherente del resultado, de esta forma se evita generar conflictos entre los participantes. Además la coordinación incluye el proceso de organización de actividades para alcanzar un resultado coherente, proporcionar mecanismos que den soluciones a los problemas que se presentan entre los miembros del equipo de trabajo y ayudar en la toma de decisiones.

Algunos aspectos que deben ser considerados en la coordinación son:

- Distribuir las actividades
- Coordinar actividades (dar orden)
- Administrar los recursos compartidos

- Integrar todas las actividades
- Armonizar la toma de decisiones para evitar conflictos o sus consecuencias

Con el objetivo de dar solución a los aspectos relacionados a la organización de actividades del grupo de trabajo y establecer un flujo de trabajo, algunos autores proponen la estandarización de procesos, los cuales que permitan: 1) distribuir las actividades entre los colaboradores, 2) establecer un orden y 3) finalmente integrar las actividades.

El control de recursos compartidos requiere una coordinación en las acciones de los colaboradores que se comunican a través de sus computadoras. Es necesario considerar la arquitectura adecuada para la transmisión de la información compartida. Whalen describe tres arquitecturas básicas utilizadas en los sistemas *Groupware*, las cuales son: *arquitectura centralizada*, *arquitectura replicada* y *arquitectura híbrida* [Whalen, 1997].

La *arquitectura centralizada* consiste en un servidor al que se conectan todos los colaboradores (también conocidos como usuarios), el servidor almacena toda la información y ejecuta los comandos de todos los usuarios en el espacio compartido. La arquitectura *replicada* se compone de varias copias de los recursos compartidos, generalmente uno por cada usuario, utiliza un proceso de comunicación de *uno a uno* para realizar cada acción y compartir los datos, sin pasar por un servidor. Finalmente la *arquitectura híbrida* utiliza cualquier combinación de estas dos arquitecturas. Existen muchos debates sobre qué arquitectura es la más apropiada en los sistemas *Groupware*, aunque la decisión siempre queda en manos del diseñador y depende principalmente del tipo de tarea que se desea realizar. En la figura 2.2 se muestra un ejemplo de como es el flujo de comunicación en la arquitectura centralizada y la arquitectura replicada.

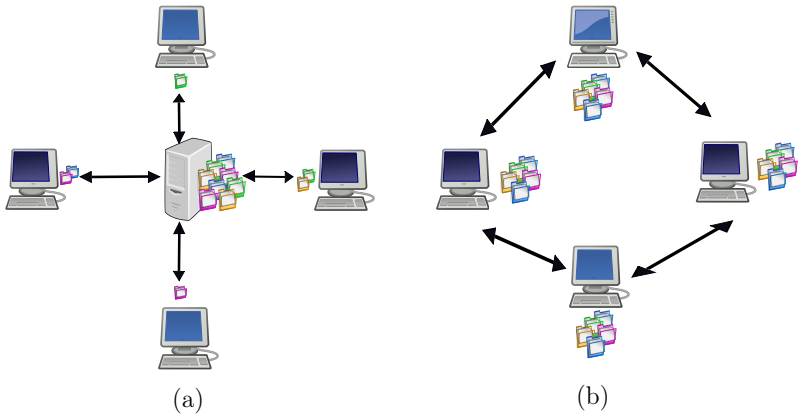


Figura 2.2: (a) Arquitectura centralizada, (b) Arquitectura replicada

Algunas de las ventajas y desventajas del uso de cada arquitectura se describen a continuación:

- *Arquitectura centralizada:* el proceso de sincronización es más sencillo, se puede tener un buen control de concurrencia en el uso de los recursos compartidos y el intercambio de datos es simple, pero la comunicación suele ser lenta y todas las acciones realizadas por los colaboradores son procesadas en un solo sitio. Esto ocasiona un cuello de botella que disminuye la capacidad de respuesta y degrada la coordinación.

- *Arquitectura replicada:* tiene un mejor tiempo de respuesta, porque se procesan las acciones en paralelo. Se utilizan varias copias que aseguran que una falla en el sistema solo afecte a un sólo participante, mientras que los demás pueden continuar trabajando. El problema es que las acciones de los participantes y los datos compartidos son difíciles de sincronizar y se requiere un proceso complicado para manejar el control de concurrencia y el intercambio de datos.

- *Arquitectura híbrida:* hace frente a estos problemas de la arquitectura centralizada y replicada, proponiendo la técnica más adecuada en cada componente del sistema *Groupware*.

La **cooperación** hace referencia a la participación intencionada y coordinada de los individuos que colaboran en el desarrollo de un proyecto común. Para facilitar la cooperación es necesario considerar el diseño de la interfaz de usuario, ya que en ocasiones es la única forma que los colaboradores tienen para interactuar con el sistema. Las interfaces *Groupware* puede ser muy simples o complejas, dependiendo de la tarea a realizar y el soporte de la tecnología utilizada en la arquitectura del sistema.

La interfaz debe ser una representación de las acciones que están haciendo los colaboradores sin afectar la actividad del usuario; utiliza el concepto de consciencia del grupo[1] para representar un entorno de trabajo compartido, que incluya retroalimentación a los colaboradores sobre quién está realizado un cambio en las regiones de interés. A menudo, las interfaces utilizan mecanismos de control de concurrencias para facilitar el acceso a los elementos de la interfaz; algunos de los mecanismos más simples son la asignación de turnos o el bloqueo exclusivo de datos para evitar conflictos.

2.1.3. Principales taxonomías de los sistemas *Groupware*

Existen diferentes formas posibles de categorizar a los sistemas *Groupware*. A continuación se expondrá las principales taxonomías:

[1] La consciencia de grupo hace referencia a la forma de pensar, el criterio y la forma de reaccionar de los individuos hacia ciertos estímulos, problemas o situaciones al pertenecer a un grupo social (familia, amigos, grupo de estudio, grupo de trabajo, etc.)

Taxonomía espacio-temporal

Los sistemas *Groupware* se pueden clasificar en función del lugar en el que se encuentre cada colaborador al interactuar con los demás colaboradores mediante el sistema; esto nos da dos espacios posibles de interacción: a) **interacción cara a cara** en donde los participantes se encuentran en el mismo lugar y b) **interacción distribuida** en donde los participantes están localizados en diferentes lugares. asimismo, el sistema puede permitir interacciones con base en el tiempo, i.e. la interacción puede ser en tiempo real o no, entonces se dice que los sistemas facilitan las interacciones de tipo **síncronas o asíncronas**. La tabla 2.1 presenta las cuatro formas posibles de categorizar estos sistemas: *interacción cara a cara*, *interacción asíncrona*, *interacción síncrona distribuida*, *interacción asíncrona distribuida*.

	Mismo tiempo	**Diferente tiempo**
Mismo lugar	Interacción cara-cara	Interacción asíncrona
Diferente lugar	Interacción distribuida síncrona	Interacción distribuida asíncrona

Tabla 2.1: Taxonomía espacio-temporal de *Groupware*

Algunos ejemplos de sistemas que pertenecen a cada uno de estos grupos son:

- **Interacción cara a cara**
  - Pantalla o mesa de visualización compartida: utilizan un proyector para visualizar e interactuar con la información presente en el sistema. Sirve como un medio de interacción entre la audiencia y el presentador.
  - Sistemas para la toma de decisiones: permiten la interacción entre un grupo de trabajo mediano (generalmente de 5 a 50 colaboradores) a través de un conjunto de computadoras y proyecciones que muestran la misma información, de esta forma estos sistemas facilitan la toma de decisiones a un grupo de trabajo.
  - Sistemas con un dispositivo de salida y múltiples dispositivos de entrada: permiten interactuar a varios usuarios sobre la misma aplicación (observando las respuestas del sistema a través de un monitor). Cada usuario interactúa a través de su propio dispositivo de entrada.
  - Oficinas inteligentes (*Roomware*): brindan un entorno de interacción que toma en cuenta la movilidad de los colaboradores y la inter-operación entre los dispositivos utilizados por los colaboradores.

  - Sistemas Multimodales: expanden la interfaz de usuario al tratar de emular a un humano que responde a las acciones que realizan los colaboradores. Para ello, estos sistemas utilizan los sentidos y habilidades del usuario tales como mecanismos de entrada y salida. Para la entrada de datos, utilizan sensores (micrófonos, pantallas táctiles, cámaras, acelerómetros y giroscopios) y el análisis de los datos que estos sensores generan.

- **Interacción asíncrona**:

  - Despliegues públicos: son anuncios de información, proveniente de varias fuentes en un lugar público.
  - Cuartos de control: la información que sirve para monitorear los turnos de trabajo.

- **Interacción síncrona distribuida**

  - Editor de texto síncrono distribuido: en esta aplicación varios usuarios localizados en diferentes lugares pueden editar un mismo documento en tiempo real.
  - Mensajería instantánea: permiten que las personas mantengan una charla en tiempo real dentro de una sala virtual mediante mensajes de texto.

- **Interacción asíncrona distribuida**

  - Correo electrónico: permite el intercambio de mensajes de forma asíncrono, sin importar que el receptor no este conectado en el momento en que se envió el mensaje este lo recibe.
  - Foros de discusión: las personas dejan comentarios respecto a un tema a discutir y pueden ser vistos en cualquier instante de tiempo.

Área de aplicación

Es posible utilizar sistemas *Groupware* en una multitud de áreas diferentes. Algunas de estas áreas son:

- **Sistema de mensajería**: soporta el intercambio asíncrono de mensajería, e.g., correo electrónico, foros de discusión, etc.

- **Editores multi-usuario**: permiten la edición y composición de un documentos por varias personas. Cuando es posible editar un mismo objeto por varios usuarios a la vez se dice que los editores son síncronos o en tiempo real, en caso contrario se dice que son asíncronos.

- **Sistemas de soporte a la toma de decisiones y salas de reunión electrónicas**: estos sistemas proporcionan facilidades para la exploración de problemas no estructurados del grupo. Las salas de reunión electrónicas incluyen herramientas para generar, organizar, categorizar y priorizar ideas; así como para la realización de votaciones.

- **Video-conferencias**: simula la experiencia de diálogo cara a cara permitiendo, en tiempo real, mantener una conversación entre usuarios que se están viendo.

- **Agentes inteligentes**: durante la sesión de trabajo cooperativo pueden intervenir participantes virtuales, también denominados agentes inteligentes, que son responsables de la realización de ciertas tareas.

- **Sistemas de coordinación**: este tipo de sistemas busca la coordinación de los esfuerzos indivuales para conseguir un objetivo mayor favoreciendo la consciencia de grupo. Estos sistemas suelen informar en qué momento se puede intervenir o realizar una acción dentro del proceso cooperativo.

- **Sistemas para la gestión de conocimiento compartido**: soportan procesos de creación, transformación, organización, búsqueda y recuperación de conocimiento.

Tarea cooperativa predominante

Grudin y Poltrock proponen un tipo de clasificación, la cual consiste en la agrupación de la tecnología con base en los tres espacios funcionales típicas de *Groupware*: comunicación, cooperación y coordinación [Cantero et al., 2001].

La mayoría de las aplicaciones pueden incluir uno o más de estos espacios funcionales, pero predomina alguna de éstos sobre las otras. La taxonomía que Grudin y Poltrock proponen para agrupar estos sistemas son:

- **Tecnologías de comunicación**
  - Correo electrónico
  - Videoconferencia
  - Difusión de video y audio[1]

- **Tecnologías de cooperación/colaboración (espacios compartidos de información)**
  - Espacios compartidos en tiempo real: permiten a las personas trabajar de forma síncrona, favoreciendo la consciencia del grupo; algunos ejemplos son:

[1]La difusión de video y audio es parecido a la emisión de televisión, esta transmisión es utilizada para la realización de conferencias en línea.

- Pizarra y aplicaciones compartidas: una pizarra compartida es un tipo de aplicación que permite a múltiples usuarios dibujar, escribir, mover el cursor, borrar, etc., de forma simultánea.
- Sistema para el soporte de la toma de decisiones y salas de reunión electrónicas
- Mundos virtuales: espacios generados por la computadora que incluyen distintas herramientas para el trabajo cooperativo.

- Espacios compartidos asíncronos: en algunos casos el trabajo cooperativo no requiere una comunicación en tiempo real o de forma simultánea, sino, sólo un espacio donde pueda ser organizada la información, donde los colaboradores puedan dejar sus contribuciones y recuperar la información creada por otros. Algunos ejemplos son:
  - Foros de discusión
  - Sistemas de gestión de documentos:

- **Tecnologías de coordinación**

  - Agenda y planificación: ayuda a los equipos de trabajo a coordinar sus actividades, sirve como sistemas de administración personal; con esta tecnología se puede consultar limitadamente las agendas personales de cada colaborador con el propósito de encontrar un día y hora conveniente para realizar una reunión de grupo.
  - Gestión de flujo de trabajo: ayuda a las organizaciones a especificar, ejecutar, monitorear y coordinar el flujo de trabajo dentro de un entorno distribuido.

2.1.4. Editores cooperativos

La aplicación que tiene un amplio campo de uso son los editores multi-usuario también conocidos como *editores cooperativos*. Algunos ejemplos de los campos de aplicación de los editores cooperativos incluyen la documentación técnica (e.g. descripción de los procesos de una organización), la documentación científica (e.g. artículos, reportes y libros), la documentación médica de los pacientes de un hospital y, en algunos casos, sirven como apoyo para la educación a distancia.

Varios autores desde los años 70 se han enfocado en el estudio de la **escritura cooperativa**. Mediante encuestas y estudios psicológicos buscan entender cómo se realiza la escritura cooperativa.

Couture y Rymer[1] distinguen dos formas en las cuales interactúan las personas al escribir: 1) *escritura en colaboración* hace referencia a cuando realmente las personas escriben

[1]Couture y Rymer en 1991 quería saber cómo se hace la escritura cooperativa en un lugar de trabajo, entrevisto a una amplia gama de personas en diferentes organizaciones. [Noël and Robert, 2004]

conjuntamente un documento; y 2) *escritura interactiva* hace referencia a cuando las personas solicitan opiniones a los demás acerca de lo que han escrito; esto redujo los casos en que realmente se utiliza la escritura cooperativa. Ede y Lunsford[1] encontraron que las personas percibían a la escritura como un acto individual por el hecho de que al escribir un documento, sólo una persona tiene acceso a los recursos (la pluma y el papel) a la vez. Es por ello que Ede y Lunsford utilizaron una definición amplia de la escritura cooperativa tomando actividades de grupo que pueden ser puestas en escrito como son: la lluvia de ideas, esquemas, toma de apuntes, planificación de la organización, redacción, revisión y edición; también se busca cual fue su contribución escrita, i.e., si la actividad dió como resultado un conjunto notas, instrucciones, reportes o materiales publicados.

Sharples et. al[2] fueron los primeros en estudiar la escritura cooperativa de los grupos de trabajo que utilizaban los procesadores de texto en las computadoras. Ellos propusieron dos estrategias para realizar un documento: 1) *secuencia longitudinal* donde se hacía la partición longitudinal del documento en secciones y cada sección era asignada a cada persona o sub-grupo en las cuales podría trabajar en paralelo; 2) *recíproco* donde las personas se reúnen para trabajar en el documento y cada miembro contribuye conforme a las reglas establecidas por ellos.

Posner y Baecker crearon una taxonomía de la escritura dividida en cuatro ejes: 1) roles, 2) actividades, 3) métodos de control de documentos y 4) estrategias de escritura. A continuación, se describe cada eje desde el punto de vista de los autores anteriormente citados según el artículo de [Noël and Robert, 2004].

1) Los **roles** hacen referencia al papel que juega cada miembro del equipo de trabajo. Posner y Baecker identificaron cuatro roles:

   - *Escritor:* quien escribe el documento,
   - *consultor:* el que ofrece información, pero no participa en la creación del documento,
   - *editor:* quien modifica directamente el documento,
   - *revisor:* el que sugiere cambios al documento sin realizarlos directamente.

2) Las **actividades** incluyen: lluvia de ideas, investigación, planeación, escritura, redacción y revisión. Un documento puede no incluir todas estas actividades y no existe un orden a seguir.

[1] Ede y Lunsford en 1990 entrevistaron a 700 personas de siete diferentes profesiones (ingeniería, química, psicología, administradores, lingüistas y administradores de servicios profesionales y escritores técnicos) para detectar como las personas perciben la escritura cooperativa [Noël and Robert, 2004].

[2] Sharples estudio como realizaban el trabajo un grupo de personas al interactuar mediante procesadores de texto (Microsoft Word, Emac), el correo electrónico y teléfono [Noël and Robert, 2004]

3) El **control de documento** describe quién y cómo se administra el documento, esto puede cambiar durante el proceso de realización del mismo. Posner y Baecker describieron cuatro tipos de métodos de edición de documentos:

   - *Centralizado:* una persona controla el documento durante todo el proceso,
   - *relevo:* una persona control el documento a la vez, pero no siempre es la misma persona,
   - *independiente:* cada persona controla la sección en la que está trabajando,
   - *compartido:* todos tienen el mismo acceso al documento.

4) Las **estrategias de escritura** describen la forma en que los miembros del equipo de trabajo cooperan juntos en la tarea de escritura; la estrategia elegida puede cambiar durante el proyecto. Posner y Baecker definieron cuatro estrategias de escritura:

   - *Escritor individual:* una persona escribe y los otros participantes juegan otros papeles en el grupo,
   - *escritores separados:* cada persona trabaja en secciones diferentes,
   - *escritores conjuntos:* los autores trabajan juntos en el texto de forma síncrona en colaboración cercana,
   - *secretario:* basado en un grupo de discusiones, un individuo escribe el documento.

Existen diferentes vertientes desde las cuales se puede abordar el diseño de los editores cooperativos. La taxonomía de Dvson (véase tabla 2.2) toma en consideración en qué se centra el sistema *Groupware*: la persona, el documento o el proceso [Cantero et al., 2001].

***Groupware* centrado en:**	**Definición**
Persona	El sistema *Groupware* gestiona localmente el trabajo de cada persona al interior de un grupo
Documento	El sistema mantiene una gestión de las tareas hechas a un documento: su realización, su consulta, actualización, etc.
Proceso	El sistema controla la conclusión de las actividades.

Tabla 2.2: Taxonomía de Dvson según el objetivo principal de *Groupware*

Como se observa en esta taxonomía, el diseño de los sistemas *Groupware* está centrado en el punto de vista que tiene el usuario sobre las tareas que realiza en el sistema. La comprensión de cómo se utilizará el sistema dentro del grupo de trabajo es crucial, por ello

para su diseño se requiere tener un análisis de la interacción que tiene cada colaborador con el sistema.

La colaboración de los miembros del grupo de trabajo depende del escenario donde se encuentra cada colaborador, como se vio en la taxonomía espacio-temporal de los sistemas Groupware. Para los sistemas que se encuentran distribuidos en espacio y tiempo, es necesario tener un mayor control del contexto de uso donde cada usuario lo utiliza, con el fin de que el sistema ofrezca mejores servicios o herramientas que den soporte al trabajo cooperativo en cuanto a los procesos de comunicación y coordinación de actividades. El diseño de los sistemas *Groupware* requiere tomar en consideración las características de las entidades dinámicas y el ambiente físico como son: tiempo, lugar, identidad, topología de la red y recursos compartidos, para proporcionar la capacidad de adaptación a los diferentes escenarios.

2.1.5. Sistemas conscientes de contexto

Para describir lo que que es un sistema consciente de contexto, es necesario conocer a qué hace referencia cada término (*consciencia* y *contexto*). De acuerdo a la investigación realizada en esta tesis de maestría, el término *consciencia* se refiere al conocimiento que el individuo tiene de sí mismo y de su entorno, mientras que el término *contexto* es definido como el entorno físico o la situación a partir de la cual se puede considerar un hecho; dicho en otras palabras, contexto es el conjunto de circunstancias que nos permiten entender una situación determinada.

En la literatura científica no existe una definición que sea ampliamente aceptada por todos los autores del término contexto debido al amplio campo de aplicación que puede tener. La definición más difundida de **consciencia de contexto** en los sistemas fue propuesta por Dey and Abownd quienes definen como contexto de un sistema a “cualquier información que pueda ser usada para caracterizar la situación de una entidad", tomando como entidad a un individuo, lugar u objeto que puede ser relevante para el sistema [Dey and Abownd, 1999].

Los **sistemas conscientes de contexto** utilizan la información recaudada para proveer información y/o servicios que son relevantes para el usuario o la aplicación en ciertas situaciones, i.e., un sistema consciente de contexto recopila información relevante de su entorno para actuar conforme a una situación determinada prestando un mejor servicio al usuario final.

La clave de este tipo de sistemas es agregar el contexto en el proceso de diseño, i.e., se debe tomar en consideración las situaciones de los usuarios que pueden estar presentes en las etapas que comprende la tarea a realizar. Para poder entender cuál es el contexto, es necesario entender el ambiente físico donde se utiliza el sistema, es por eso que el con-

texto juega un papel importante en áreas de investigación como en el Cómputo Ubicuo, en el cual los sistemas deben considerar la información contextual como el tiempo, la ubicación, los recursos disponibles y los usuarios que representan los aspectos del mundo físico, para procesar esa información e integrarla a las actividades de la vida cotidiana [Haake et al., 2010].

La información contextual depende de la situación del usuario o del sistema en un punto determinado del tiempo pero, por naturaleza, esta información suele estar incompleta lo que requiere una interpolación de los hechos que forman parte del proceso, por lo cual se habla de un "contexto de uso" y no simplemente del contexto [Coutaz and Calvary, 2007]. Algunos autores dividen el contexto de uso en diferentes categorías que nos permiten conocer las características a considerar en los sistemas conscientes de contexto.

Schilit et al. proponen dividir el contexto de uso en tres categorías: contexto computacional (e.g. conectividad, ancho de banda, interfaces, etc), contexto de usuario (e.g. perfil, ubicación, etc) y contexto físico (e.g. temperatura, nivel de luz, cantidad de sonido, etc) [Schilit et al., 1994]. Una división más especifica para los sistemas computacionales está dada por Coutaz y Calvary que divide al contexto de uso en: usuario, plataforma y ambiente [Coutaz and Calvary, 2007], donde:

1. El *usuario* denota a la persona que interactúa con el sistema. El modelo del usuario incluye el perfil, la idiosincrasia, las tareas y actividades de la persona.
2. La *plataforma* se refiere a los recursos de hardware y software disponibles con las cuales el usuario interactúa con el sistema. El modelo de la plataforma puede ser descrito en términos de sensores, redes de comunicación y recursos que unen el ambiente físico con el mundo digital.
3. El *ambiente* describe las condiciones físicas y sociales donde la interacción toma lugar. El modelo del entorno incluye la ubicación del usuario (e.g. oficina, casa, lugar público), reglas sociales, actividades y condiciones de luz y sonido.

De acuerdo a Crowley et al., el contexto de uso puede describirse en función de una red de roles y relaciones [Crowley et al., 2002]. Diferentes configuraciones de roles y relaciones corresponden a diferentes situaciones contextuales. Coutaz and Rey coinciden en señalar que el contexto no puede existir de forma aislada, sino que debe ser definido con respecto a entidades particulares como el usuario, el sistema o la tarea a realizar [Coutaz and Rey, 2002]. Ellos definen al contexto de uso como una composición de múltiples situaciones en un cierto periodo de tiempo.

Scmidth presenta un mapa taxonómico sobre las características de los sistemas que incluyen la consciencia del contexto de uso [Dey and Mankoff, 2005]. A continuación se presenta la taxonomía:

I. Percepción contextual es la habilidad de detectar información contextual y presentarla al usuario para aumentar su capacidad sensorial.

II. Adaptación contextual es la habilidad de ejecutar o modificar un servicio de forma automática con base en el contexto actual, i.e. las acciones disparadas por el contexto.

III. Descubrimientos de los recursos contextuales, los cuales permiten que la aplicación sea consciente del contexto dependiendo de su ubicación y de la exploración de los recursos o servicios disponibles que son relevantes al usuario.

IV. Crecimiento contextual es la habilidad de relacionar datos digitales al contexto del usuario, presentando la información y/o servicios, dependiendo dónde y con quién se encuentre el usuario.

2.2. Trabajos relacionados

Fueron considerados para esta investigación algunos sistemas *Groupware* comerciales y algunos de uso académico. A continuación se presenta una breve descripción de cada uno.

2.2.1. Dropbox

Dropbox[1] es un servicio de alojamiento de archivos multi-plataforma en la nube[2] operado por la compañía Dropbox. El servicio permite a los usuarios almacenar y sincronizar archivos en línea entre computadoras y con otros usuarios. Existen versiones gratuitas y de pago, cada una de las cuales cuenta con capacidades de almacenamiento que van desde los 2GB a 18GB de forma gratuita y 500GB a 1TB con pago mensual.

Para acceder a Dropbox es necesario registrarse en su página web, utilizando un correo electrónico. Al registrarse se obtiene espacio de almacenamiento en la nube. Dropbox permite organizar los archivos en carpetas. Así mismo cuenta con aplicaciones de escritorio para las diferentes plataformas existentes (Linux, Mac OSX, Windows, Android, IOS).

En la aplicación de escritorio se genera una copia de los documentos que le pertenecen a cada usuario, utilizando una técnica de replicación, cuya actualización se realiza automáticamente vía Internet mientras el usuario se encuentre en línea.

Esta aplicación es útil debido a que se puede acceder a los documentos en cualquier momento y tener una única versión del mismo. Además, dicha aplicación permite compartir

[1] www.dropbox.com

[2] La computación en la nube es un paradigma que permite ofrecer servicios de cómputo y espacio de almacenamiento a través de Internet

los documentos con otros usuarios a través de invitación vía correo electrónico.

En la figura 2.3 a) se muestra que Dropbox cuenta con una aplicación web y una aplicación de escritorio. Ambas aplicaciones presentan un inicio de sesión, cuyo usuario es registrado previamente por su correo electrónico y la sesión identifica al usuario. En la figura 2.3 muestra la aplicación web, la cual presenta un menú de opciones de Dropbox de lado izquierdo que permite al usuario conocer las actividades realizadas, quiénes están compartiendo sus documentos y una guía para comenzar a utilizar la aplicación. Así mismo presenta la lista de archivos y carpetas que están almacenadas en la nube. En la parte superior de esta sección existe un menú de opciones de edición de la lista de archivos, el cual permite, crear, subir, compartir, eliminar y buscar diferentes carpetas en el directorio de archivos. En la figura 2.4 b) se muestra la aplicación de escritorio, la cual establece una carpeta en el sistema de archivos, cuyo contenido será sincronizado con Dropbox, i.e., se realiza la copia de los archivos de la nube perteneciente al usuario registrado (proceso de replicación) y las opciones sobre la misma.

Así mismo en la figura 2.4 c) se presenta las opciones que permiten compartir una carpeta desde la aplicación web y conocer quiénes tienen acceso a la misma. También ofrece realizar una invitación de los usuarios a través del envío de un correo electrónico.

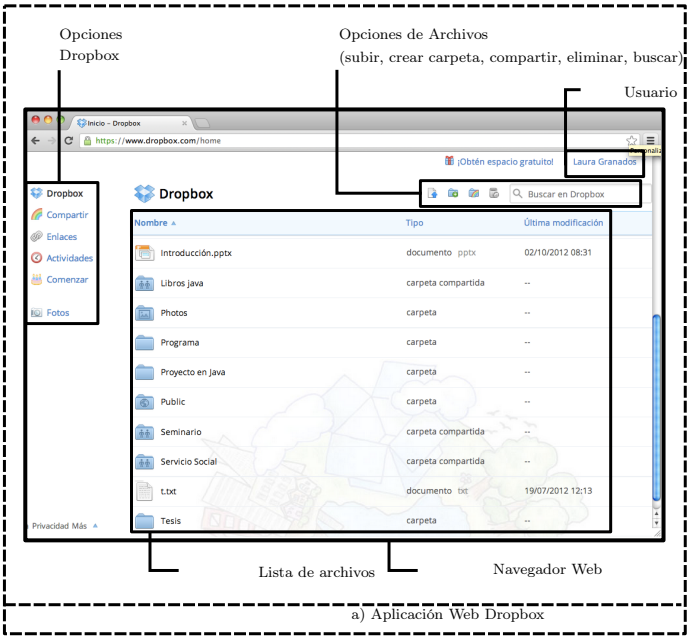


Figura 2.3: Dropbox

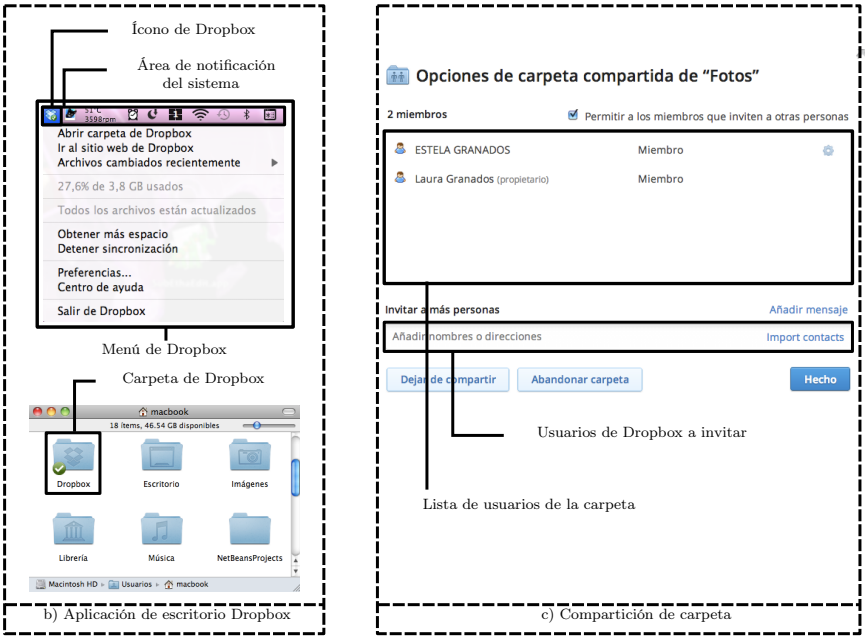


Figura 2.4: Dropbox

2.2.2. iFolder

iFolder[1] es una aplicación de código abierto desarrollada por Novell Inc. que permite compartir archivos a través de Internet entre diferentes plataformas. Los archivos están disponibles en línea y fuera de línea. Cualquier cambio realizado en una carpeta de iFolder es actualizado automáticamente en el servidor y sincronizado a las demás computadoras que estén utilizando iFolder. iFolder puede usarse para realizar copias de seguridad, lo cual permite acceder y gestionar los archivos personales. A diferencia de Dropbox, esta aplicación requiere tener un servidor de almacenamiento detrás de un firewall.

iFolder requiere la instalación de iFolder-Server en un servidor de la empresa (o del usuario). iFolder-Server funciona sobre Apache[2] en las plataformas de Linux y Windows. Se requiere dar permisos a los usuarios para que puedan tener acceso a la carpeta compartida, además requiere en el servidor un firewall para proteger la información.

Novell iFolder-Sever permite administrar los recursos del servidor, verificar el acceso de los clientes al servidor, almacenar las carpetas y documentos compartidos, etc.

La aplicación del cliente iFolder se puede ver como un explorador de carpetas remotas, la cual permite al usuario establecer un enlace de comunicación con el servidor que contiene la aplicación Novell iFolder-Server, mediante un inicio de sesión. El usuario puede subir nuevas carpetas y tener acceso a las carpetas compartidas; la sincronización con el servi-

[1] http://www.novell.com/products/openenterpriseserver/features/online-file-storage-ifolder.html
[2] Apache es un servidor web que provee funcionalidades a los desarrolladores web.

dor es programable por cada usuario; la convención de sincronización de archivos está al inicio de sesión y posteriormente cada cinco minutos.

Además de permitir el acceso desde cualquier ubicación, permite compartir las carpetas con diferentes usuarios y establecer permisos de solo lectura , de lectura y escritura o control total [1].

En la figura 2.5 a) se muestra la configuración de iFolder-Server, misma que se realiza a través de un navegador web, en el servidor se establecen los permisos de acceso a los usuarios de la aplicación; en la figura 2.5 b) se muestra la aplicación de escritorio, la cual permite administrar los archivos y carpetas de cada usuario, así como establecer los permisos a los usuarios con quienes se comparten.

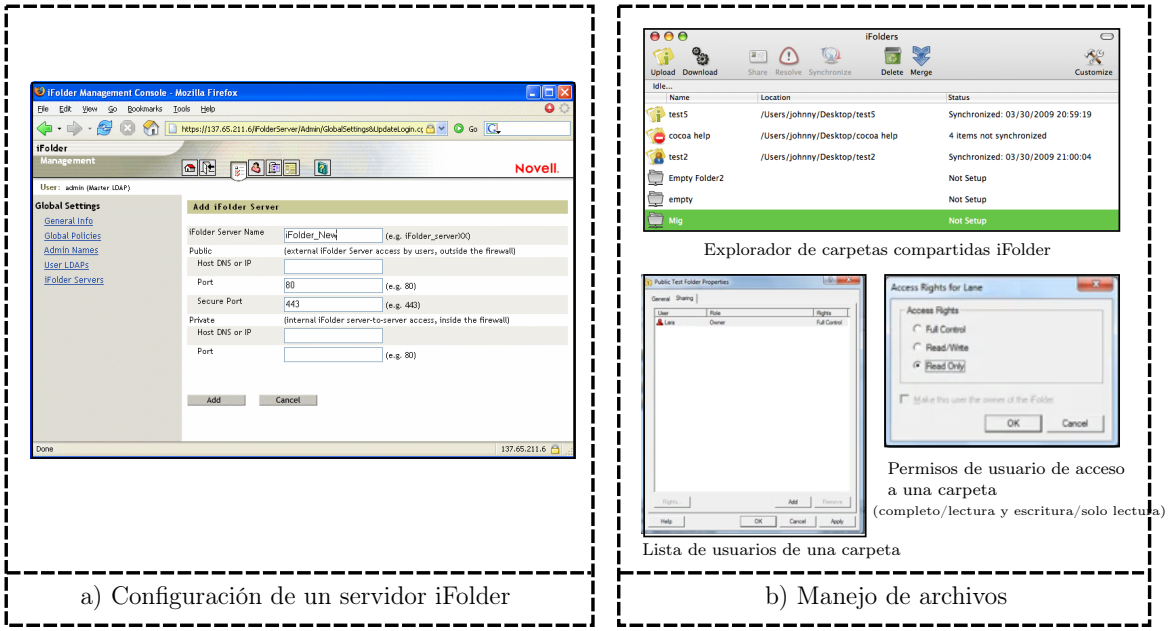


Figura 2.5: iFolder

2.2.3. Abiword

Abiword [2] comenzó a ser desarrollado por la empresa SourceGear en el año de 1998 y tenía como objetivo crear un paquete de software de oficina libre y multi-plataforma. Sin embargo, sólo se finalizó el desarrollo del procesador de texto. Abiword utiliza el servicio web AbiCollab.net para almacenar y compartir los archivos a un grupo de personas; así mismo permite la importación y exportación de documentos en su formato nativo a XML, RTF, HTML, Microsoft Word, LaTeXy OpenDocument. Su principal uso es la edición de

[1] iFolder funciona sobre Mono otra aplicación de Novell Inc.
[2] http://www.abisource.com

documentos en tiempo real.

Para compartir los documentos se debe ingresar a la aplicación *AbiCollab* en la web, esta aplicación requiere un correo electrónico para registrarse y solicita un inicio de sesión. AbiCollab permite almacenar todos los documentos creador por Abiword en su formato original, es decir, con la extensión .abicollab. Esta aplicación web cuenta con tres secciones, la que presenta los documentos guardados, los amigos que están conectados y los grupos de trabajo que se tienen. En la sección de documentos se presenta una lista de los archivos cargados y la opción de crear un nuevo documento o actualizar un documento. En la lista de archivos se tiene la opción de abrir, ver, compartir, etiquetar, ver la historia, exportarlo a un diferente formato y eliminarlo.

El editor Abiword cuenta con el estándar de los editores de documentos, el cual nos permite modificar el tipo y tamaño de fuente, los estilos de letras y la alineación de párrafo. Cuenta con los menús: Archivo, Edición, Vista, Insertar, Formato, Herramientas, Tablas, Ventana y Ayuda, con el diseño de los menús el usuario está familiarizado en la edición de un documento. En las figuras 2.6 y 2.7 se muestran ambas aplicaciones que permiten la edición cooperativa de documentos con Abiword y AbiCollab.

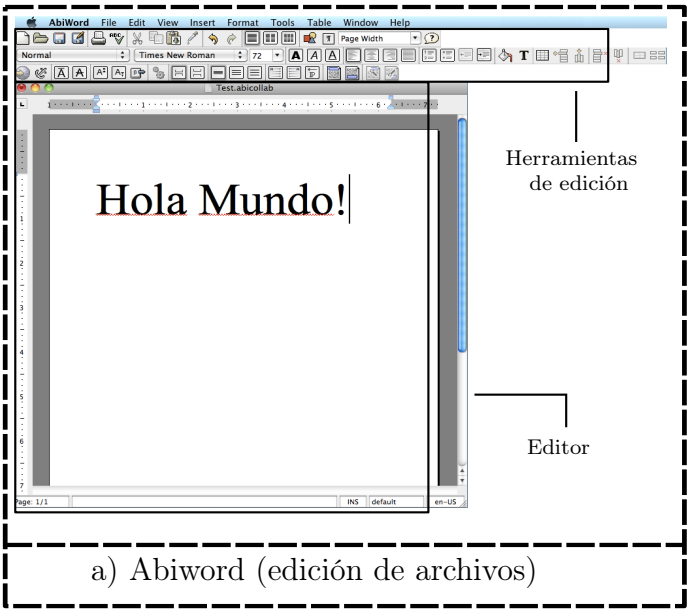


Figura 2.6: a) Abicollab

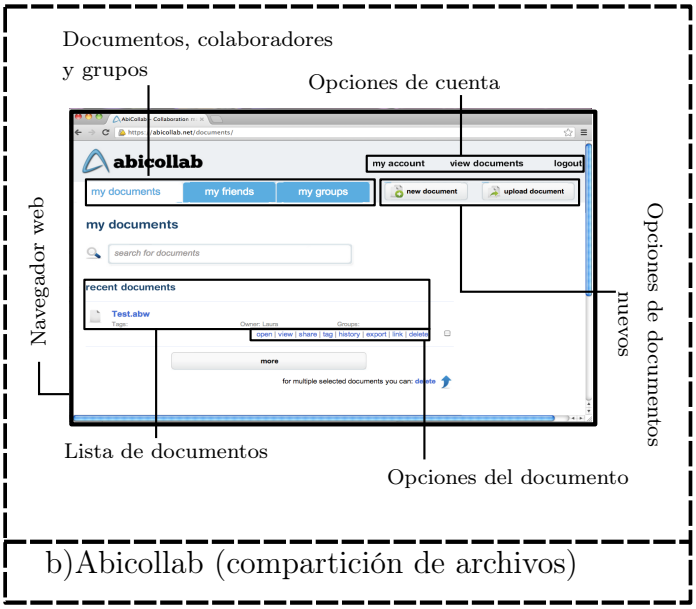


Figura 2.7: b) AbiWord

2.2.4. Gobby

Gobby [1] es un software de colaboración en tiempo real, el cual sigue una arquitectura cliente-servidor que da soporte a varios documentos en una sola sesión. La sincronización de los documentos se realiza a petición del usuario y para su acceso a la aplicación se requiere el uso de una contraseña. Cuenta con una sala de conversación (Chat) en texto para la comunicación de los usuarios. Los usuarios pueden elegir un color para resaltar el texto que han escrito en el documento, también el sistema proporciona un proceso que resalta la sintaxis para los lenguajes de programación básica.

Dentro de las opciones que nos presenta Gobby está la lista de usuarios conectados y la lista de documentos compartidos, tal como se muestra en la figura 2.8. La diferencia con Gobby es que principalmente es utilizado para archivos que no requieren un estilo en la fuente del documento, i.e. documentos en texto plano.

[1] http://gobby.0x539.de/trac

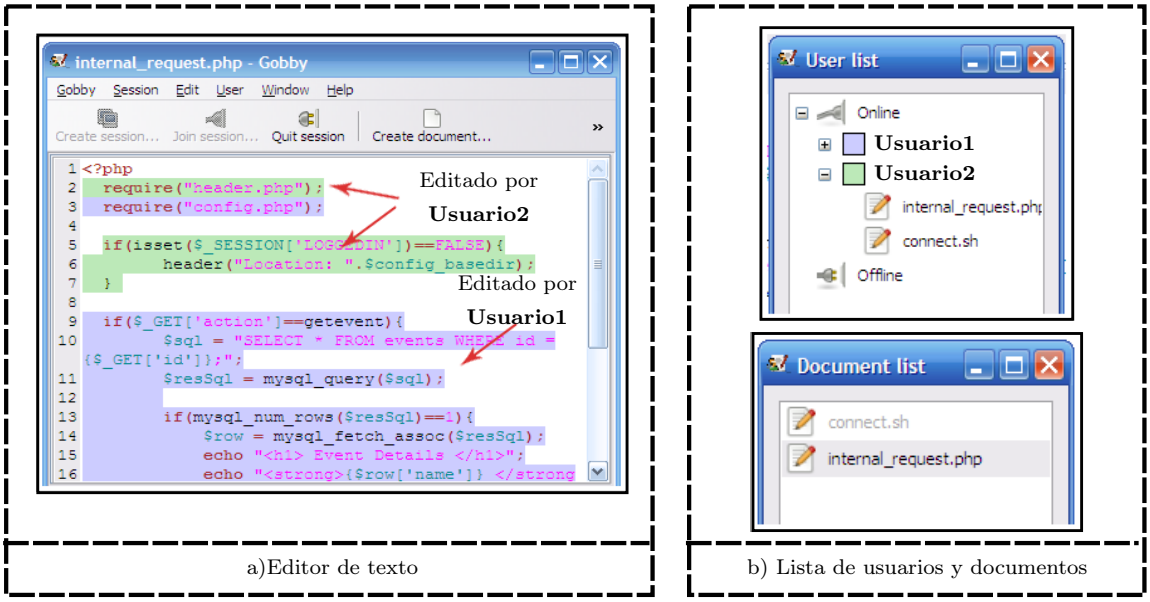


Figura 2.8: Gobby

2.2.5. SubEthaEdit

SubEthaEdit [1] (anteriormente conocido como Hydra) es un potente editor de texto colaborativo diseñado para Mac OS X que utiliza las características de Bonjour, el servicio de descubrimiento de Apple.

A diferencia de los sistemas de control de versiones como Subversion o CVS, que guardan una copia consistente de un documento, SubEthaEdit ofrece colaboración con la posibilidad de editar el mismo documento en tiempo real, junto con todos los miembros de un grupo. SubEthaEdit es especialmente adecuado para programación extrema[2] y la creación de notas colaborativas en conferencias.

SubEthaEdit permite la detección de usuarios de forma automática siempre y cuando se encuentren en la misma red. Para compartir un documento es necesario anunciarlo en la aplicación, con ello se informa a los demás usuarios en la red que documento está disponible para visualizarlo y cuál es el estado del mismo (bloqueado, sólo lectura o lectura y escritura). El editor ofrece un resaltado de sintaxis del texto ingresado por cada colaborador, así como informa en qué linea del documento está trabajando cada uno.

SubEthaEdit maneja un control de anunciar los documentos en el que está trabajando cada usuario, un control de acceso a los documentos, localización automática de usuarios, tal como se presenta en la figura 2.9

[1] http://www.codingmonkeys.de/subethaedit/

[2] Programación extrema es una metodología de desarrollo de la ingeniería de software que pone énfasis en la adaptabilidad que en la previsibilidad.

Dentro de las facilidades que ofrece para compartir la información es que permite exportar el documento a un formato HTML, el cual puede incluir la información del documento mediante la selección de opciones que se presenta al exportar el documento: color de sintaxis, marcas de cambio, acerca de los escritores, fecha actual e información particular de los participantes.

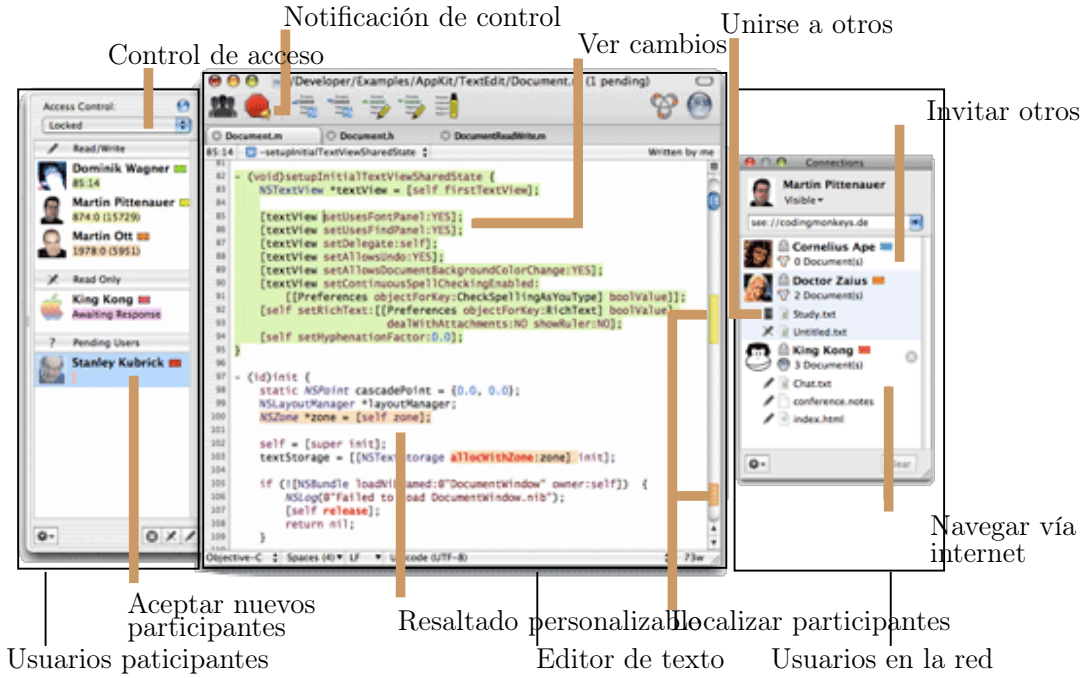


Figura 2.9: SubEthaEdit

2.2.6. Microsoft SharePoint Workspace

SharePoint [1] se refiere a una combinación de las tecnologías de Microsoft que facilitan la colaboración dentro de una organización, anteriormente era conocido con el nombre de Microsoft Office Grove. Sharepoint Services[2] (véase Figura 2.10) ofrece funciones de colaboración basadas en el explorador de web, módulos de administración de procesos, módulos de búsqueda y una plataforma para la administración del documento. Funciona en conjunto con SharePoint Server de Microsoft Office Project Server y Portal, de modo que los usuarios pueden crear y administrar sitios web de colaboración para ponerlos a disposición de toda la organización.

La funcionalidad de SharePoint es: administrar reuniones o tareas; así como almacenar, organizar y compartir documentos mediante una carpeta compartida (utilizando el ex-

[1] http://office.microsoft.com/es-es/sharepoint-workspace
[2] Imagen obtenida del manual de usuario

plorador de Windows). Usa credenciales para autenticación. Los documentos se pueden utilizar con o sin conexión a la red.

Se utiliza para generar formularios y compartirlos, lo cual facilita el llenado posterior, como si fuese una plantilla de llenado. Utiliza el servicio de mensajería Microsoft Lync para poner en contacto a los colaboradores. La forma de compartir la información es a petición del usuario. Al estar soportado por Windows, se requiere una red segura de comunicación para que la información esté protegida de ataques.

En la figura 2.10 se puede observar que, desde el navegador de Internet Explorer, se puede acceder a la lista de documentos compartidos.

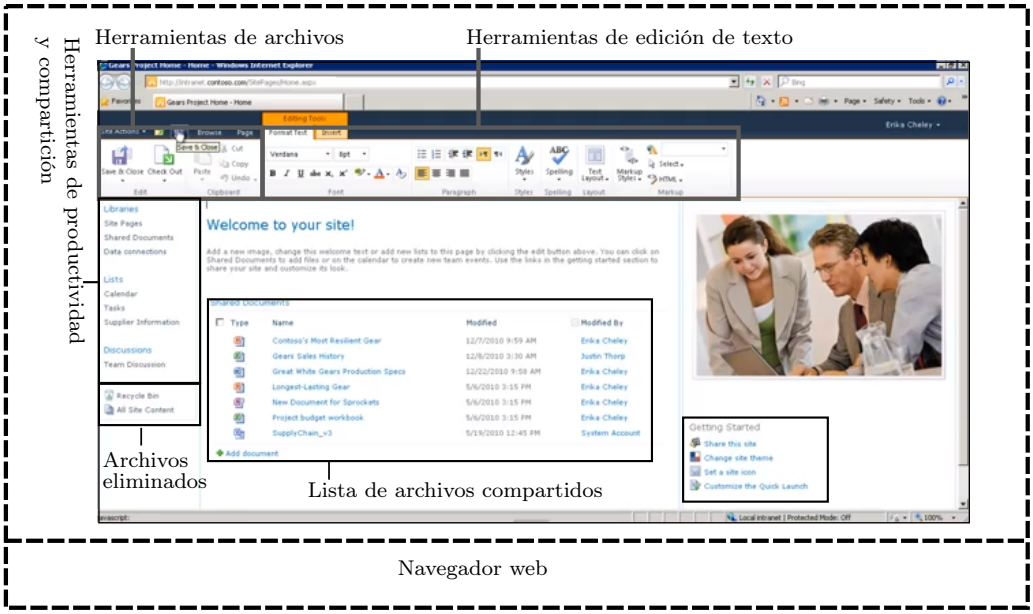


Figura 2.10: SharePoint

2.2.7. Google Docs

Google Docs [1] herramienta ofrecida por Google que combina las características de Writely y Spreadsheets, con el objetivo de ofrecer una paquetería de oficina que permita crear y editar documentos en línea, así como la colaboración de múltiples usuarios en los documentos. Los documentos se almacenan automáticamente en los servidores de Google para evitar la perdida de datos, y la aplicación de Google Docs mantiene un historial de versiones, aunque no los clasifica por los cambios realizados.

La compañia Google ofrece un conjunto de herramientas que permiten el trabajo cooperativo desde la nube. El servicio que ellos ofrecen no necesita tener instalado un programa

[1] www.google.com/google-d-s/tour1.html

en particular para que los usuarios puedan acceder y editar la información, simplemente es necesario un navegador web y una cuenta de correo de Gmail (servicio de correo electrónico de Google) para poder tener acceso a la información cargada en la nube.

A partir del navegador, Google Docs permite crear, subir, compartir y editar documento en línea, los cambios realizados por cada colaborador se ven reflejados a nivel carácter. Además cuenta con servicios de comunicación vía mensaje de texto, video-llamada o llamada, con su nuevo servicio Hangout, el cual permite compartir imágenes de forma instantánea.

Acepta la mayoría de los formatos de archivo comunes, como DOC, XLS, ODT, ODS, RTF, CSV, PPT, etc. La principal desventaja es que al encontrarse en la nube, sin Internet no se puede acceder a la información.

En las figuras 2.11 y 2.12 se muestran la lista de archivos subidos a Google Docs y las opciones de compartir estos archivos, así como la edición de un documento desde el navegador.

a) Visualizando presentación de PowerPoint en GoogleDocs

Figura 2.11: a) Ejemplo de uso de GoogleDocs

e

b) Interfaz web de documentos GoogleDocs

c) Opciones de compartición de archivos

Figura 2.12: b) Interfaz Web de documentos GoogleDocs y c) Opciones para compartir archivos

2.2.8. ShowDocument

ShowDocument [1] fue desarrollada por HBR Labs en 2007. Es una aplicación web que permite a varios usuarios llevar a cabo reuniones electrónicas, subir y compartir información y revisar documentos desde lugares remotos. Los usuarios pueden colaborar y revisar documentos en tiempo real, con anotaciones y texto visible a todos los usuarios, además permite la co-edición del mismo. Es posible dibujar y escribir en una pizarra virtual y transmitir un video vía Youtube el cual puede ser visualizado por todos los miembros del grupo de trabajo.

Está diseñado para el apoyo de los sistemas de *e-learning* pues proporciona herramientas que facilitan la transmisión de información entre todos los participantes de forma simultánea para un aprendizaje conjunto. La versión gratuita está limitada a tres personas y sesiones de 30 minutos, y una versión de paga permite un uso ilimitado de tiempo.

En la figura 2.13 se muestran todas las aplicaciones que nos permite utilizar conjuntamente la aplicación.

[1] http://www.showdocument.com

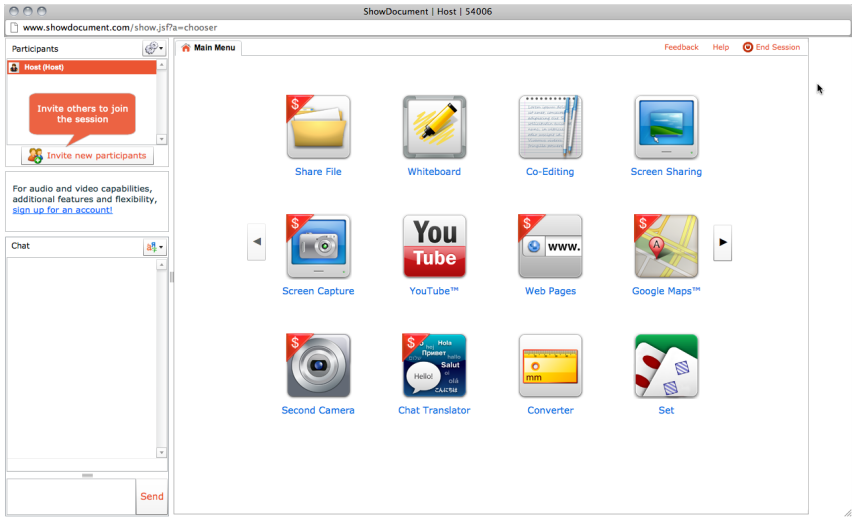


Figura 2.13: Showdocument

2.2.9. CodoxWord

CodoxWord [1] es un SDK para Microsoft Word que convierte el procesador de texto en un editor cooperativo en tiempo real. Combina los cambios realizados por cada participante al instante sin necesidad de actualizar el documento. Notifica los cambios realizados por cada participante, utilizando un color de resaltado de texto y permite compartir el documento desde un repositorio web o bajo una arquitectura P2P.

En la figura 2.14 se muestra como el SDK se integra automáticamente a MicrosoftWord y lo convierte en un editor cooperativo.

2.2.10. CollabEd

CollabEd [Granville and Hickey, 2009] es una plataforma de los sistemas de colaboración de edición. Es un plug-in para los editores de programación existentes más utilizados (Eclipse, Netbeans, jEdit, etc) y un programa de dibujo (DrawSWF). Una de las características que se destacan es la posibilidad de guardar las sesiones de colaboración para su posterior reproducción con las estadísticas de edición específicas del usuario. En la figura 2.15 se muestra cómo se ven los cambios realizados por cada uno de los participantes.

[1] www.codoxware.com/codoxword

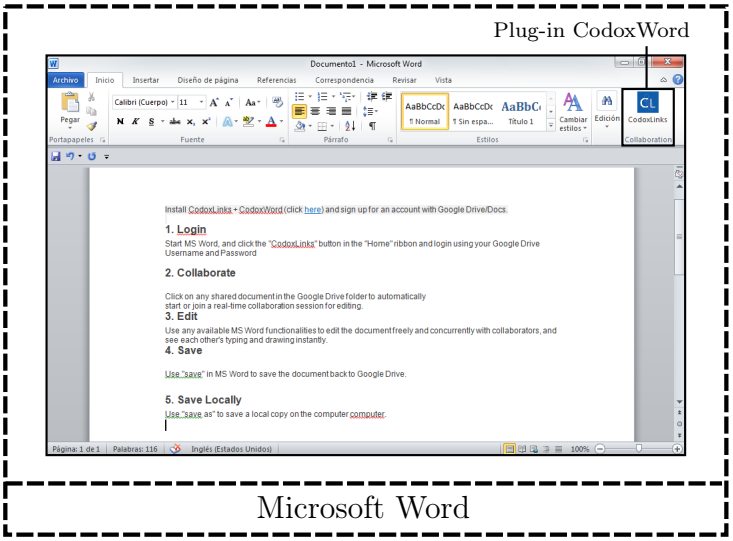


Figura 2.14: CodoxWord

2.2.11. CoCoDoc

CoCoDoc [Hofte and Opendoc, 1997] fue una iniciativa para el soporte del trabajo en grupo, con base en OpenDoc y CORBA. CoCoDoc proporciona un marco para el uso de los editores de cooperativos. También soporta el desarrollo de un nuevo conjunto de editores cooperativos con facilidades de colaboración flexibles, facilitando una migración gradual hacia los entornos de edición colaborativa, ricos en edición y soporte cooperativo. CoCoDoc fue parte del proyecto Platinum, un proyecto de unión de investigación en aplicaciones multimedia de CSCW sobre redes de banda ancha.

2.2.12. CoopDraw

CoopDraw [Ramstein, 1993] es la primera aplicación interactiva desarrollada para el entorno COOP. Se trata de un editor de gráficos estructurado que permite a varios usuarios interactuar simultáneamente en un documento. COOP es un proyecto de investigación llevado a cabo por CM (Centro de Información Innovación Technologies) de los sistemas CSCW. Su objetivo es producir un ambiente propicio para el desarrollo de software colaborativo síncrono.

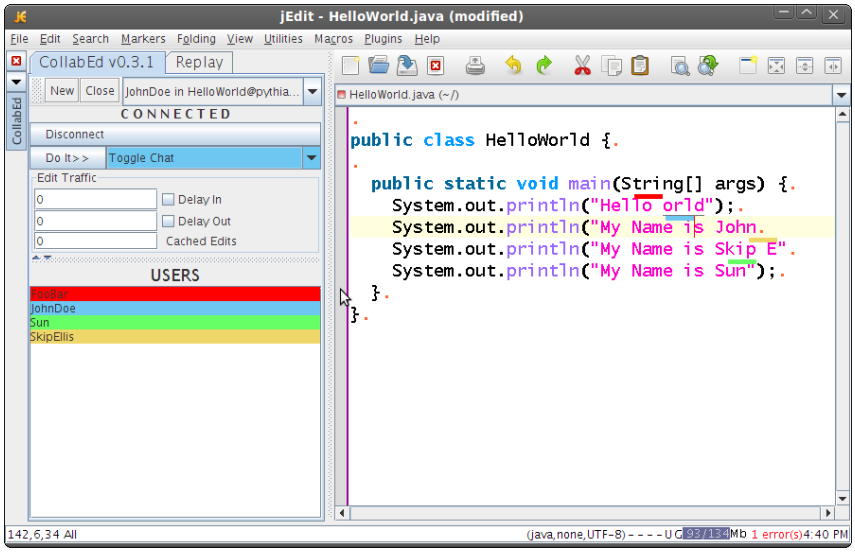


Figura 2.15: CollabEd

2.3. Criterios de comparación

Durante la investigación se seleccionaron los siguientes criterios que nos permiten comparar a los sistemas antes mencionados. Los criterios que se seleccionaron son: funcionalidad, arquitectura, enfoque, sincronización, plataforma, participación de los usuarios, control de concurrencia y proceso de notificación. A continuación se describen cada uno de los criterios.

2.3.1. Criterio de funcionalidad

Se utilizó la taxonomía del nivel de aplicación de los sistemas *Groupware* para conocer las funcionalidades que ofrecen los trabajos relacionados y el entorno propuesto.

- **Mensajería instantánea**: proporciona a los usuarios una forma de comunicarse a través de mensajes simultáneos.
- **Conferencias y reuniones electrónicas**: ofrecen a los usuarios un canal de comunicación compartido, mediante una interfaz o espacio de trabajo donde pueden hablar y compartir información de forma simultánea.
- **Sistemas de soporte a la toma de decisión**: facilitan la exploración de problemas no estructurados por parte del grupo y mejora la toma de decisiones.
- **Administración de documentos**: proporciona características tales como indexación, búsqueda y distribución de documentos entre varios usuarios.

- **Edición cooperativa de documentos**: incluyen a la administración de documentos y extiende sus características, con el fin de proporcionar un control de versiones y gestión de los cambios realizados al documento.

2.3.2. Criterios arquitectónicos

Los criterios de arquitectura de los sistemas *Groupware* definen dónde y cómo se gestiona la colaboración de los miembros del grupo de trabajo.

- **Arquitectura centralizada**: la colaboración es administrada por un servidor central; todos los datos se intercambian a través de un punto central de acceso. Esta arquitectura está relacionada directamente con la arquitectura cliente-servidor.
- **Arquitectura replicada**: la colaboración es administrada en todos los nodos en la red; los datos y la información se intercambian de nodo a nodo. Esta arquitectura esta relacionada con la arquitectura peer-to-peer.
- **Arquitectura híbrida**: es una combinación de ambas arquitecturas anteriores, donde existen super-nodos que procesan la información intercambiada por los usuarios.

2.3.3. Criterio de enfoque

Las actividades que pueden realizarse en el sistema están definidas bajo un enfoque de colaboración particular.

- **Centrado al usuario**: el proceso de colaboración se centran en el usuario, i.e., el sistema crea un canal de comunicación entre los colaboradores con el fin de facilitar la realización de la tarea.
- **Centrado en la tarea**: el proceso de colaboración se centra en la realización de la tarea específica; típicamente se encarga del almacenamiento, la estructura y los cambios transcurridos en el tiempo de realización. El canal de comunicación es transparente.
- **Centrado en el espacio de trabajo**: puede ser visto como una extensión al espacio de trabajo de los sistemas *Groupware*, i.e. si se puede almacenar el estado del área de trabajo y permite a los colaboradores conocer los cambios realizados en la información compartida por el sistema.

2.3.4. Criterio con base a la sincronización

Los criterios temporales definen las restricciones impuestas para la colaboración con respecto al tiempo de actualización de cambios.

- **Síncrona**: la actualización se hace al instante en que se realiza el cambio, también conocida como tiempo real
- **Asíncrona**: la actualización se realiza a petición del usuario
- **Mixto**: la actualización depende de los mecanismos que utilice cada colaborador al momento de compartir la información, puede ser en tiempo real o de forma asíncrona.

2.3.5. Criterio con base en la plataforma

Define la compatibilidad de la aplicación con base en la plataforma de software.

- **Basado en la plataforma del sistema operativo**: la colaboración sólo puede ocurrir en los nodos que comparten el mismo sistema operativo.
- **Plataformas basadas en navegadores Web**: la colaboración puede ocurrir a través de cualquier navegador Web.
- **Independiente de la plataforma (Multi-plataforma)**: la colaboración puede ocurrir en múltiples plataformas.
- **Plataformas móviles**: la colaboración puede extenderse a dispositivos móviles y de mano.

2.3.6. Criterio con base en la participación del usuario

Define el nivel de participación por parte del usuario para obtener ventajas proporcionadas por el software cooperativo.

- Participación **alta**: significa que los usuarios se ven obligados a trabajar con una interfaz diferente a lo habitual, con el fin de acceder a las funciones de colaboración. Esto es típico de los ambientes de trabajo compartido.
- Participación **media** significa que los usuarios pueden trabajar con sus interfaces de usuario normales y sólo tendrán que ejecutar comandos de colaboración en un momento dado.
- Participación **baja**: significa que el usuario sólo está implicado en la creación del entorno de colaboración y luego puede seguir trabajando en la tarea asignada como si no estuviera colaborando en el equipo de trabajo. Todas las funciones de colaboración son automatizadas y destinadas a ser transparentes para el usuario

2.3.7. Criterio de manejo del control de concurrencia

Describe el mecanismo de control de concurrencia para la utilización de recursos compartidos en el sistema y para evitar conflictos. Los mecanismos más utilizados son:

- **Por turnos**: asigna turnos de utilización de los recursos compartidos, organiza el acceso con el uso de un calendario que establece quien puede utilizar el recurso en una fecha y hora indicada.
- **Bloqueo exclusivo**: cuando el usuario utiliza un recurso, el sistema bloquea dicho recurso a los demás usuarios y lo desbloquea hasta que el primer usuario deje de utilizarlo.
- **Algoritmo de control de concurrencia**: utiliza algún algoritmo especifico para el evitar conflictos y mantener la consistencia de los datos.

2.3.8. Criterio con base en la notificación de cambios

Describe si el sistema utiliza un sistema de notificación de los cambios realizados por los demás colaboradores y cómo lo realiza.

- **Tiempo real**: se notifica automáticamente los cambios realizados por otros usuarios.
- **Asíncrona**: se notifica bajo petición del usuario los cambios realizados por otros usuarios.

2.4. Comparación de los trabajos relacionados

De acuerdo a los criterios antes mencionados, se realizó una comparación de los trabajos relacionados.

Como se observó en los trabajos relacionados (véase tabla 2.3) algunos sistemas se especializan sólo en ofrecer una funcionalidad en específico, e.g., las aplicaciones iFolder y Dropbox se especializan solo en administrar documentos compartidos, pero no ofrecen un mecanismo de comunicación o de toma de decisiones; mientras que otras aplicaciones utilizan una combinación de las funcionalidades, con el fin de apoyar el trabajo de los colaboradores. La aplicación más completa encontrada es Google Docs, la cual ofrece un mecanismo de comunicación utilizando salas de mensajería instantánea o conferencias electrónicas y un mecanismo para la administración de documentos. Además permite la edición de un documento a través de un navegador web. Algo que se observó al hacer la comparación bajo el criterio de funcionalidad, es que ninguno ofrece un mecanismo que de soporte a la toma de decisiones.

Criterios		Sistemas *Groupware*										
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord
Funcionalidad	Mensajería instantánea	-	-	-	X	-	X	X	X	-	-	-
	Conferencias y reuniones electrónicas	-	-	-	-	-	-	X	X	-	-	-
	Sistema de soporte a la toma de decisiones	-	-	-	-	-	-	-	-	-	-	-
	Administración de documentos	X	X	-	-	-	-	-	X	-	-	-
	Edición cooperativa	-	-	X	X	X	X	X	X	X	X	X

Tabla 2.3: Criterio de funcionalidad

El criterio de arquitectura ofrece propuestas de cuál va a ser la arquitectura a utilizar en el entorno cooperativo. La mayoría de los sistemas se basan en un sistema centralizado o híbrido, ya que cuentan con un servidor que controla los cambios en los documentos, pero a su vez mantiene una copia del documento en la terminal de los clientes conectados. Sólo CodoxWord permite la comunicación de forma centralizada o bajo replicación, utilizando una topología peer-to-peer aunque no puede clasificarse como híbrido (véase tabla 2.4).

Criterios		Sistemas *Groupware*										
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord
Arquitectónico	Centralizada	X	-	-	X	X	X	-	-	X	X	X
	Replicado	-	-	-	-	-	-	-	-	-	-	X
	Híbrido	-	X	X	-	-	-	X	X	-	-	-

Tabla 2.4: Criterio de arquitectura

El criterio de enfoque permitió conocer cuál es el enfoque que tienen los sistemas actuales y cuáles son los que es necesario explorar en el entorno propuesto. Aunque la mayoría está centrado en el usuario y la forma en que realiza la tarea, pocos toman en consideración el espacio de trabajo para ofrecer servicios que ayuden a los colaboradores. En la tabla 2.5 se muestra el enfoque de cada sistema.

Criterios		Sistemas *Groupware*										
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord
Enfoque	Centrado al usuario	-	-	-	-	-	-	X	X	-	-	-
	Centrado a la tarea	X	X	X	X	X	X	X	X	X	X	X
	Centrado al espacio de trabajo	-	-	-	-	-	-	-	-	-	-	-

Tabla 2.5: Criterio de Enfoque

El criterio con base al tipo de sincronización se refiere a cómo se realiza la actualización de la información. La mayoría de las aplicaciones realizan la actualización bajo petición del usuario y es difícil conocer los cambios realizados por los otros participantes, lo cual provoca problemas de coherencia y malos entendidos entre los usuarios. Algunos otros sí refrescan los cambios realizados automáticamente, pero en ocasiones causa confusión a los usuarios ver todas las actualizaciones realizadas por otros usuarios. En la tabla 2.6, se muestra la comparación con base en el tiempo de los sistemas estudiados.

Criterios		Sistemas *Groupware*										
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord
En base al tiempo	Síncrono	X	X	X	-	-	X	X	-	X	X	-
	Asíncrono	-	-	-	X	X	X	-	-	X	-	-
	Mixto	-	-	-	-	-	-	-	-	-	-	-

Tabla 2.6: Criterio con base en el tiempo

El criterio de plataforma sirvió para diseñar el entorno cooperativo conforme a la plataforma más utilizada en los entornos cooperativos, aunque la tendencia actual es basarse en los navegadores web, con el fin de generar una aplicación que pueda ser utilizada en cualquier plataforma. Algunos de los sistemas encontrados son soportados por uno o varios sistemas operativos (véase tabla 2.7).

Criterios		Sistemas *Groupware*										
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord
Plataforma	Sistema Operativo	X	-	X	X	-	-	-	-	X	-	X
	Navegador Web	-	-	-	-	-	-	-	X	X	-	-
	Multi-plataforma	-	X	-	-	X	X	-	-	-	X	-
	Móvil	-	-	-	-	-	-	-	-	-	-	-

Tabla 2.7: Criterio de plataforma

En cuanto al criterio de la participación del usuario se consideran tres niveles que llegan a ofrecer los sistemas actuales, con el fin de proporcionar un nivel igual a la mayoría, como se ve en la tabla 2.8

Criterios		Sistemas *Groupware*										
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord
Participación del usuario	Alto	X	X	-	X	-	-	-	-	-	-	-
	Medio	-	-	X	-	X	X	-	-	X	X	X
	Bajo	-	-	-	-	-	-	-	X	X	-	-

Tabla 2.8: Criterio de participación del usuario

El criterio de manejo de control de concurrencia sirve para establecer el mecanismo a utilizar, haciendo una comparación de los mecanismos más utilizados en los sistemas *Groupware* estudiados (véase tabla 2.9).

Criterios		Sistemas *Groupware*										
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord
Control de concurrencia	Serialización	X	X	-	-	-	-	-	-	-	-	-
	Bloqueo exclusivo	-	-	X	-	-	X	X	-	-	-	-
	Algoritmo específico	-	-	-	-	-	-	-	-	-	-	-

Tabla 2.9: Criterio de control de concurrencia

El criterio de notificación permite conocer los mecanismos que utilizan las aplicaciones para detectar los cambios realizados por los participantes. Como ejemplos de estos mecanismos, se puede mencionar la utilización de notas o comentarios dentro del documento, con el fin de expresar ideas al respecto o el resaltado de texto ingresado por cada colaborador, permitiendo la edición simultánea (véase tabla 2.10).

		Sistemas *Groupware*										
	Criterios	iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord
Notificación	Notas o comentarios	-	-	X	X	-	-	X	X	X	-	X
	Resaltado de texto	-	-	-	-	X	X	-	-	-	-	-

Tabla 2.10: Criterio de control de notificación

Capítulo 3
Análisis y diseño

Este capítulo muestra el análisis y diseño del entono cooperativo *Misho*. Se inicia con el estudio de los problemas a los que se enfrentan los miembros de un grupo en la edición cooperativa de un documento, con el fin de definir las características funcionales que se incluyeron en el diseño de *Misho*. El capítulo se divide en cinco secciones: la sección 3.1 muestra los problemas que se enfrentan los grupos de trabajo al editar cooperativamente documentos e imágenes; la sección 3.2 refleja los prototipos analizados que sirvieron como guía para el diseño de *Misho*; la sección 3.3 realiza un análisis de las técnicas y tecnologías de los sistemas computacionales, con el fin de utilizarlas al dar solución a los problemas antes descritos; la sección 3.4 análisis de *Misho*; y finalmente, la sección 3.5 detalla el diseño de *Misho*.

3.1. Problemática existente en la edición cooperativa con las herramientas actuales

De forma similar a la de un grupo de personas que se reúnen y se complementan entre ellos para realizar una tarea común, se busca que los sistemas *Groupware* ofrezcan mecanismos que imiten este ambiente de trabajo y a la vez aumenten la eficacia de los miembros del grupo en cuanto a la comunicación, producción y coordinación de las actividades que realizan.

Los miembros del grupo de trabajo al interactuar por medio de sus computadoras, requieren mecanismos que les permitan: 1) realizar la edición de un documento, 2) compartir la información a los demás miembros del grupo, 3) comunicarse entre sí, y 4) coordinar las actividades del mismo.

Como se explicó en el capítulo del estado del arte, las aplicaciones para la edición cooperativa de documentos llegan a incluir uno o dos de los tres espacios funcionales de los

sistemas *Groupware*, dejando como responsabilidad de los miembros del grupo de trabajo los espacios restantes. Esto ocasiona que los editores cooperativos actuales no logren recrear el ambiente del grupo de trabajo, al no proporcionar la consciencia de que se está trabajando en grupo.

Al no cubrir las necesidades del grupo del trabajo, los participantes se ven obligados a buscar aplicaciones adicionales (no especializadas), que si bien ayudan en la realización de la tarea, no son un medio eficaz; e.g. en la figura 3.1 (página 52) se puede observar que dos miembros del grupo utilizan de un lector de documentos (Foxit Reader) para visualizar el documento y una ventana de chat (gTalk) para poder discutir una sección del mismo, estas aplicaciones permiten trabajar sobre la revisión de un documento de forma conjunta a dos o más personas, a pesar de no ser aplicaciones cooperativas.

Cuando se redacta un documento (individual o conjuntamente) suele ser necesario incluir imágenes que se relacionen al texto escrito, por lo que los colaboradores llegan a utilizar tanto un editor de texto, como un editor de imágenes. El utilizar varias aplicaciones a la vez, ocasiona que los colaboradores se distraigan de las actividades que tienen asignadas y pierdan tiempo al cambiar de una aplicación a otra. Debido a la variedad de aplicaciones que pueden ser utilizadas por el grupo de trabajo, los participantes pueden elegir entre usar aplicaciones mono usuario o multiusuario. A continuación, se describen los escenarios de cada caso:

Escenario 1: uso de aplicaciones de edición monousuario

La edición del documento se realiza por un participante a la vez y requiere un mecanismo para compartir el documento. Cuando el grupo de trabajo utiliza este tipo de aplicaciones, es necesario que previamente se decidan las actividades y el rol que va a desempeñar cada miembro del grupo dentro de las secciones del documento. Las aportaciones que realiza cada participante son realizadas de manera individual y posteriormente compartidas mediante el envío de un correo electrónico o dado personalmente en una reunión de trabajo a los demás colaboradores, esto provoca que los cambios realizados al documento sean conocidos hasta que son compartidos.

Si el grupo de trabajo no tiene una buena coordinación de las actividades que realiza cada participante, puede ocurrir que dos o más colaboradores trabajen en la misma sección del documento a la vez. Esto provoca inconsistencias en el documento al momento de integrar los cambios y que en ocasiones se lleguen a perder los cambios previamente realizados por algún otro colaborador. Para evitar tanto el problema de perdida de información, como mantener una coherencia en la misma, se requiere que uno de los miembros del grupo sea el encargado de incorporar todas las aportaciones realizadas por los demás colaboradores, lo cual es una tarea laboriosa y que requiere mucho tiempo.

Escenario 2: uso de aplicaciones de edición multiusuario

Los editores multiusuario, también son conocidos como editores cooperativos, son aplicaciones que permiten la edición y composición de un documento por varias personas utilizando una red de computadoras. Como se vió en el capítulo de estado del arte existen diferentes aplicaciones que permiten la edición cooperativa de documentos, la mayoría de estas aplicaciones ofrecen editar documentos de texto plano y un mecanismo para compartir la información entre los colaboradores conectados a la aplicación.

Los editores cooperativos actuales ofrecen tres mecanismos que sirven como un medio de comunicación entre los colaboradores del grupo de trabajo al editar un documento de forma conjunta, estos mecanismos son: 1) agregar comentarios dentro del documento, 2) introducir notas en el documento y 3) enviar mensajes de texto mediante una sala de mensajería de instantánea.

Aunque los tres mecanismos son de gran ayuda cuando se está revisando un documento, presentan una limitante a los colaboradores, debido a que la interacción entre ellos se realiza por medio de sus computadoras y no pueden emplear gestos que usualmente se utilizarían al interactuar de forma presencial, e.g. señalar una sección de texto dentro del documento para un análisis o discusión, este procedimiento es muy sencillo si se tiene una interacción cara a cara, pero puede ser muy complicado si los participantes se encuentran a distancia, dado que el colaborador debe informar a sus compañeros el documento en el que está la sección a discutir y la ubicación exacta del texto, además indicar el contenido del mismo para que los demás colaboradores puedan identificar a qué texto se está refiriendo y poder entonces realizar el análisis de la información; esto provoca que los colaboradores pierdan tiempo al enfocarse en la ubicación del texto más que en la idea principal a discutir. La figura 3.1 muestra un ejemplo de como se realiza el análisis de una sección de texto de un documento utilizando una sala de mensajería instantánea por dos colaboradores.

Otro problema al que se enfrentan los colaboradores es el proceso de sincronización, i.e. cómo y cuándo se actualizan los cambios realizados por los colaboradores en el documento común. Cada aplicación ofrece un mecanismo de sincronización diferente, algunos deciden utilizar una sincronización síncrona, dónde todos los cambios realizados por los colaboradores se ven reflejados instantáneamente en el editor de cada colaborador; mientras que otros esperan una petición del usuario (sincronización asíncrona) para poder compartir la información con los demás colaboradores y obtener los cambios actuales. En este proceso de sincronización, la información se almacena localmente en el editor de cada colaborador para ser compartida posteriormente.

Ambos procesos de sincronización pueden llegar a representar un problema a los colaboradores debido a que el proceso de sincronización depende principalmente del rol de cada colaborador en la edición del documento. La sincronización síncrona es útil, si un cola-

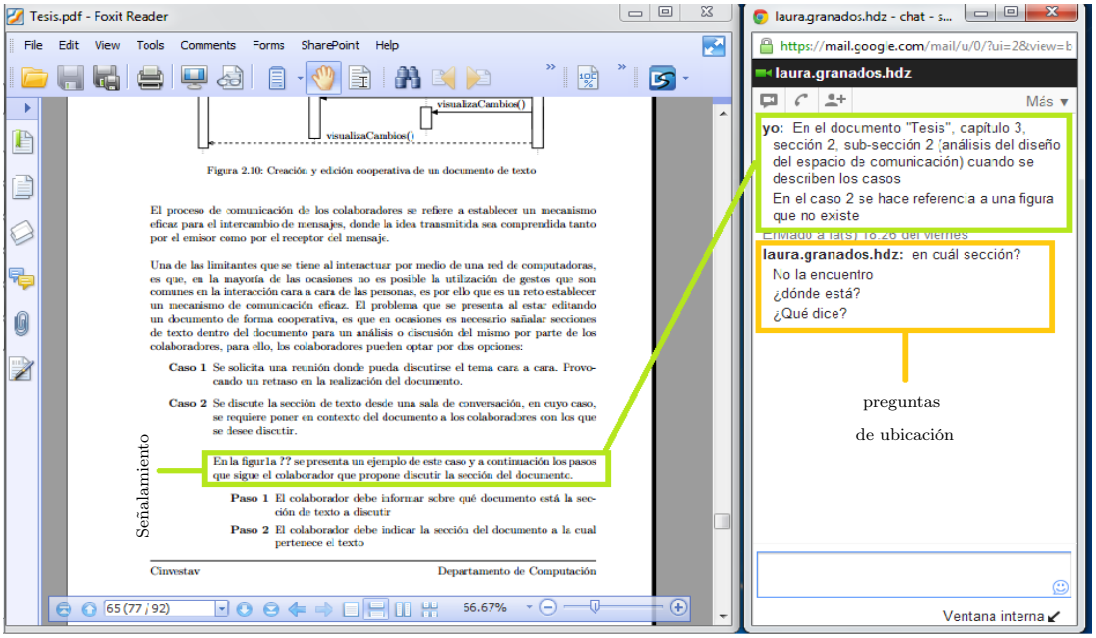


Figura 3.1: Conversación en la que se análiza una sección de texto de un documento

borador tiene el rol de revisor, dado que el colaborador necesita monitorear todo lo que los demás colaboradores están realizando, sin embargo, para el colaborador que tiene un rol de escritor, el recibir todas las notificaciones de los cambios realizados por los demás colaboradores, puede ocasionar una distracción en la ejecución de la tarea asignada. Por otra parte, si se utiliza una sincronización asíncrona, pueden llegar a perderse cambios realizados previamente por otros colaboradores.

Para evitar conflictos en la utilización de recursos compartidos y mantener la coherencia en la información del documento, es necesario incluir una técnica de control de concurrencias. Los problemas de concurrencia ocurren principalmente cuando un recurso es compartido por dos o más entidades a la vez, tomando en consideración que una entidad puede ser un persona, un sistema o aplicación. E.g. cuando dos o más colaboradores están editando la misma figura, puede existir un conflicto de interés entre ellos al tratar de editar la misma propiedad de dicha figura al mismo tiempo, e.e. los colaboradores cambian de color el relleno de la figura.

La figura 3.2 presenta el problema de concurrencia existente en el ejemplo anterior, tomando en consideración que se tienen dos colaboradores editando una misma figura. En el caso (a) el colaborador C_1 modifica la propiedad borde de la figura a un color rojo, al no existir otra operación sobre esta propiedad de la figura, no se ocasiona conflicto alguno. En el caso (b) las operaciones realizadas por los colaboradores ocasionan un conflicto de interés, ya que ambas afectan la misma propiedad de la figura: el relleno. El sistema puede elegir una acción a realizar de las cuatro posibilidades existentes: 1) respetar el último

cambio en la propiedad de la figura: el relleno de la figura será amarillo; 2) ignorar la última petición y tomar el primer cambio recibido: el relleno de la figura será verde, 3) ignorar ambas operaciones y dejar el relleno de la figura en su estado original, y finalmente 4) dejar que los colaboradores resuelvan el problema.

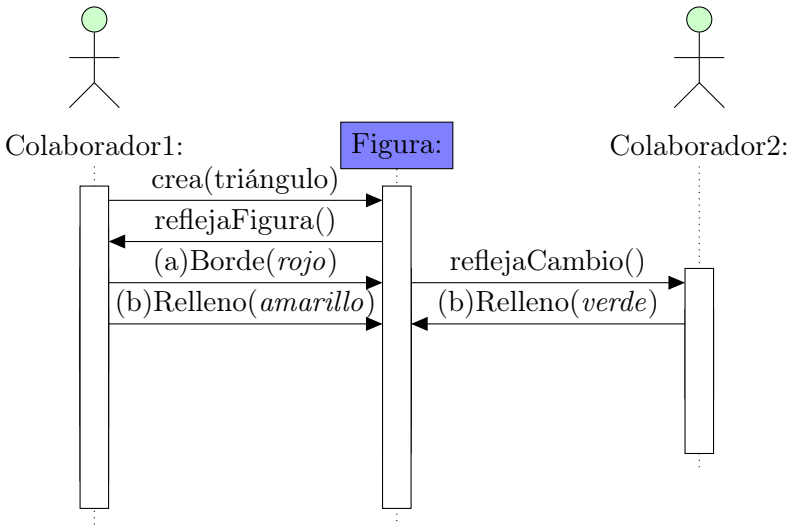


Figura 3.2: Problema de concurrencia en la edición de una figura

3.2. Antecedentes de *Misho*

Para explorar las técnicas y metodologías que pueden ser implementadas en el desarrollo de editores cooperativos, en el Departamento de Computación del Centro de Investigaciones y de Estudios Avanzados del Instituto Politécnico Nacional se desarrollaron tres prototipos. El primer prototipo que se analizó es un editor de texto que permite la edición de un documento por turnos entre dos clientes; el segundo prototipo es un pizarrón cooperativo que permite la edición de una imagen y finalmente, el tercer prototipo es una sala de mensajería instantánea que permite la comunicación entre los colaboradores del grupo de trabajo. A continuación se describen a detalle estos prototipos.

3.2.1. Prototipo de editor de texto

El editor de texto es una aplicación desarrollada en Java conformada por un servidor y un cliente, los cuales se conectan entre sí para permitir la edición de un documento HTML de forma conjunta a dos colaboradores. Dentro de las funcionalidades que ofrece se encuentran: cambiar el tipo, color y tamaño de fuente, así como los estilos y la alineación

del párrafo. En la figura 3.3 se muestran las opciones que presenta el editor de texto al usuario para personalizar el formato de texto del documento.

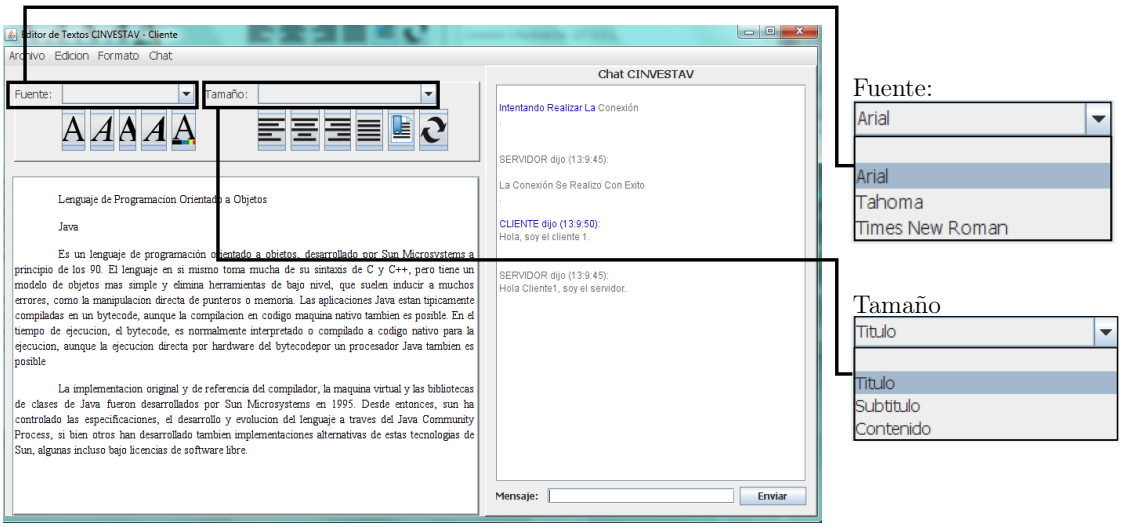


Figura 3.3: Herramientas de formato de texto

El servidor administra un directorio compartido que almacena los documentos creados y editados en el editor de texto por cada colaborador. El cliente accede al directorio compartido, abre el documento y carga la información del mismo en el editor de texto para que pueda ser visualizada por el colaborador.

El editor de texto, permite la edición del documento por turnos, i.e. solo un colaborador puede editar la información del documento a la vez. El servidor otorga el permiso de edición al primer colaborador que lo solicita, bloqueando la edición al segundo colaborador hasta que el primero indique que terminó de editarlo y viceversa.

Los cambios realizados por el colaborador que tiene el permiso de edición, se almacenan localmente en su editor de texto. Para poder compartir los datos con el otro colaborador, se requiere guardar los cambios en el documento y enviar la solicitud de actualización al servidor para que éste envíe una notificación al otro colaborador para que actualice el documento.

El editor de texto incluye una sala de mensajería instantánea que permite la comunicación entre los dos colaboradores. Esta sala de mensajería facilita el trabajo de los colaboradores al poder señalar una sección de texto dentro del documento y compartirla (copiando el texto seleccionado) como un mensaje en la sala de mensajería instantánea. Esta funcionalidad se basa en buscar la primera ocurrencia del texto a compartir dentro del documento y resaltarlo al cambiar el fondo del mismo en el editor. Esto es un problema si el texto a compartir se repite en más de una ocasión, ya que puede resaltarse el texto en una sección

anterior del documento.

En la figura 3.4 se muestra el funcionamiento del editor de texto con los dos clientes (cliente A, y cliente B) los cuales se conectan al servidor y tienen acceso a la carpeta compartida que contiene el documento que están editando los colaboradores.

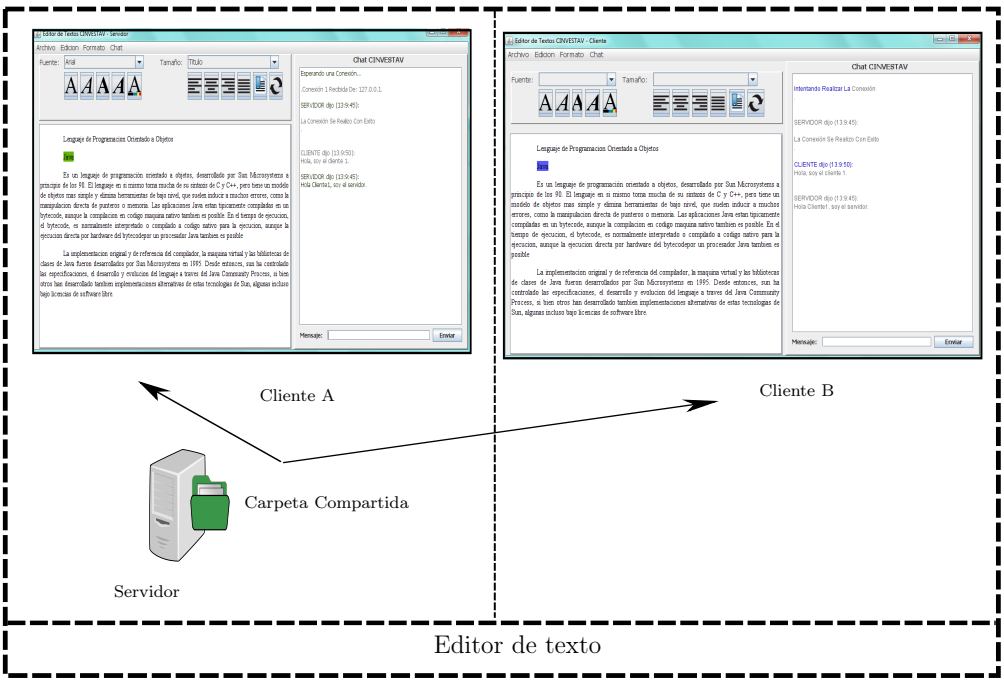


Figura 3.4: Prototipo de editor de texto

3.2.2. Prototipo de editor de imágenes

El editor de imágenes es una aplicación desarrollada en Java que permite la edición cooperativa de imágenes basada en una arquitectura cliente-servidor. El servidor del prototipo se encarga de la administración y comunicación de los clientes utilizando la tecnología de actualización *Push*[1].

La interfaz de usuario del editor de imágenes está integrada por dos componentes: una barra de herramientas, que contiene las figuras que se pueden pintar en el pizarrón (texto, línea recta, triángulo, cuadrado, círculo), así como las herramientas de edición de las propiedades de las figuras (borde y relleno); y también contiene un lienzo, que es el área de edición en la que se puede dibujar.

[1]La tecnología *Push* describe un estilo de comunicaciones sobre Internet donde la petición de una transacción se origina en el servidor.

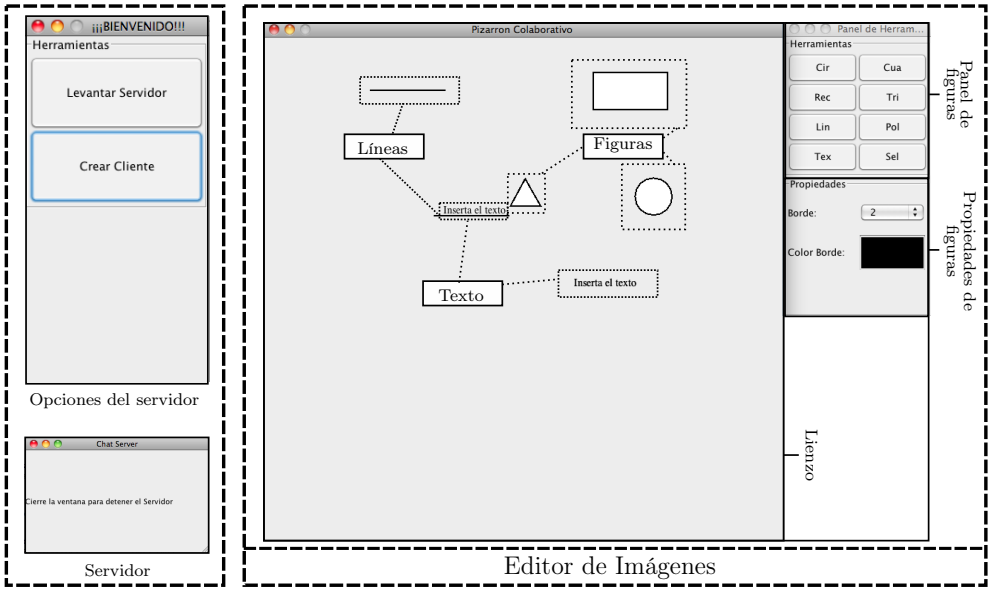


Figura 3.5: Editor de imágenes

Las opciones de la barra de herramientas que pueden realizarse dentro del lienzo son: insertar, seleccionar y editar figura. Al hacer un click izquierdo con el ratón dentro del lienzo, se realiza la opción seleccionada sobre el lienzo, i.e. en las coordenadas del ratón con respecto al lienzo.

En este prototipo, los cambios realizados por cada colaborador C en el editor de imágenes se envían al servidor, el cual se encarga de actualizar la imagen a todos los colaboradores conectados. Este prototipo almacena los cambios realizados en una sesión. Cuando un colaborador inicia sesión, el servidor le envía automáticamente el estado actual del lienzo (véase figura 3.6).

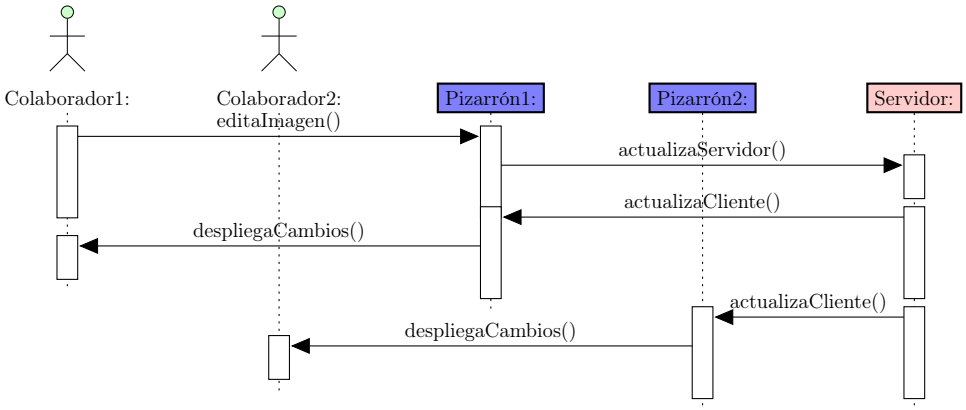


Figura 3.6: Creación y edición cooperativa de una imagen

3.2.3. Prototipo de sala de mensajería instantánea

La sala de mensajería es un módulo de comunicación que permite compartir ideas entre los colaboradores. Cuando un colaborador inicia sesión, se identifica con un nombre, mismo que es utilizado para identificar los mensajes realizados por él, en la sala de mensajería instantánea. En la figura 3.7 se presenta la interfaz gráfica de la sala de mensajería la cual esta incluida en el prototipo del editor de imágenes descrito anteriormente.

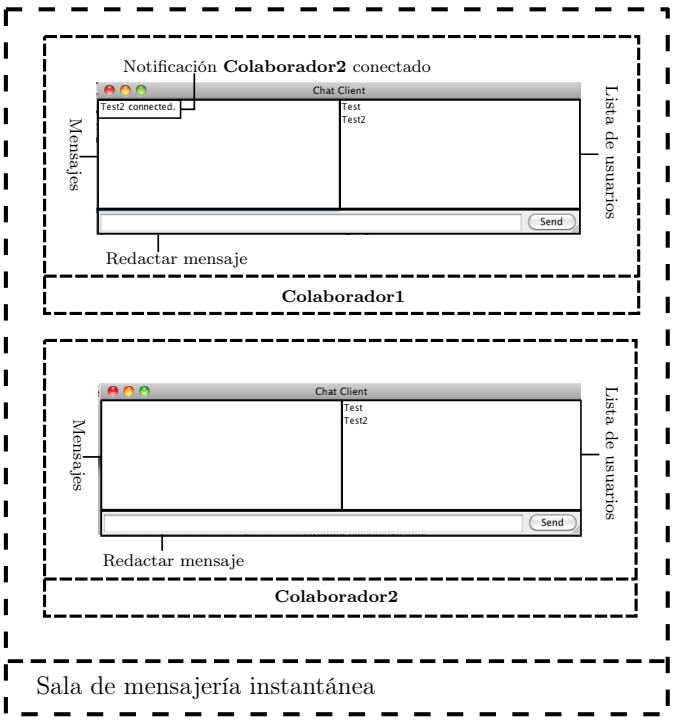


Figura 3.7: Editor de imágenes

3.3. Análisis de las técnicas y tecnologías de desarrollo

Como se mencionó en el capítulo de introducción (página 1), los sistemas *Groupware* utilizan los avances tecnológicos de las disciplinas de los sistemas computacionales para cumplir sus objetivos. En esta sección se realiza un análisis de las técnicas y tecnologías que pueden ser aplicadas en el desarrollo de *Misho*, con la finalidad elegir las que ofrecen una mayor contribución para el entorno propuesto. Se estudiaron: 1) las técnicas de sincronización, 2) las técnicas de control de concurrencia, 3) los lenguajes de programación y finalmente 4) las tecnologías de comunicación en red.

3.3.1. Técnicas de sincronización

Los sistemas distribuidos utilizan técnicas de sincronización para coordinar los cambios en los recursos compartidos. La sincronización es un proceso por el cual dos o mas componentes intercambian datos o información.

Las técnicas utilizadas para la sincronización de datos pueden clasificarse en: sincronización síncrona, sincronización periódica, sincronización por número de transacciones y sincronización asíncrona. A continuación se describe cada una de estás técnicas.
técnicas:

Técnica síncrona o de sincronización automática

De acuerdo con esta técnica, la actualización de los cambios en la información (realizados por un colaborador) son informados a los demás colaboradores conectados de forma instantánea.

El problema que se presenta en este tipo de sincronización es que los colaboradores pueden distraerse con facilidad debido al alto número de notificaciones que se pueden mostrar en el espacio de producción. Además, en ocasiones, la idea de estar siendo monitoreados llega a intimidar a los colaboradores, ya que todos los cambios que realice el colaborador son vistos por sus compañeros.

Este tipo de sincronización es de gran ayuda en algunas etapas de la edición colaborativa de documentos, como la lluvia de ideas y la planeación de la estructura del documento.

Técnica asíncrona o bajo petición del usuario

En esta técnica, la actualización de los cambios en la información se realiza a petición del usuario, i.e. el colaborador decide cuándo compartir los cambios que ha hecho

y cuándo recibir los cambios realizados por los demás colaboradores. El problema en este tipo de sincronización es que el colaborador se aísla del grupo de trabajo durante la edición del documento.

Este tipo de sincronización es de gran ayuda cuando los colaboradores tienen tareas específicas asignadas y no existe una codependencia en sus actividades.

Técnica de sincronización periódica o por tiempo

En esta técnica, la actualización de cambios en la información se realiza en periodos de tiempo determinados. El período de tiempo se establece dependiendo la tarea que realiza el grupof de trabajo, debido a que un periodo muy pequeño ocasionaría los mismos problemas que en la sincronización automática, mientras que un periodo de tiempo muy largo ocasionaría los mismos problemas que en la sincronización a petición del usuario (sincronización asíncrona).

Este tipo de sincronización es de gran ayuda cuando en el proceso de notificación de cambios realizados por otros colaboradores, en la sección del documento que el colaborador esta editando es de forma asíncrona.

Técnica de sincronización por número de transacciones

En esta técnica, se almacenan cierto número de cambios (realizados por el colaborador) antes de compartirlos a los demás colaboradores. El problema de este tipo de sincronización es que la latencia de notificación depende del número de transacciones y del tiempo que toma a un colaborador alcanzar la cuota de cambios. El número de transacciones se denomina como *granularidad de sincronización.*

Este tipo de sincronización es de gran ayuda al compartir los cambios realizados en el documento de texto, ya que se puede permitir al colaborador elegir el tipo de granularidad con la cual se actualizan los cambios, i.e. cada colaborador puede elegir si recibe los cambios a nivel: carácter, palabra (conjunto de caracteres) o párrafo (conjunto de palabras).

Para el diseño de *Misho* se optó por ofrecer a los colaboradores el uso de más de un tipo de sincronización sobre los espacios funcionales de producción y comunicación, dependiendo la granularidad de las transacciones que pueden realizarse en el editor de texto, el pizarrón y la sala de mensajería instantánea que los componen, i.e. elegir entre las técnicas de sincronización síncrona, asíncrona y por número de transacciones.

3.3.2. Técnicas de control de concurrencia

Las técnicas de control de concurrencia permiten gestionar el acceso a los recursos compartidos, a través del establecimiento de un conjunto de reglas a seguir para evitar inconsistencias en la información.

A continuación, se describen brevemente las técnicas de control de transacciones existentes: utilización de marcas de tiempo, técnicas multiversión y protocolos optimistas de validación, que ofrecen reglas en las transacciones que permiten tener un control de concurrencia de los recursos compartidos. Además se describe la técnica de bloqueo, la cual permite tener un control de acceso al recurso compartido.

Utilización de marcas de tiempo

Esta técnica es utilizada para la ejecución correcta de las transacciones de un proceso, estableciendo una marca de tiempo (identificador único) a cada transacción que se ejecuta en él; requiere el uso de un contador y del reloj del sistema, además se basa en el almacenamiento de la última transacción ejecutada y la transacción activa. Las marcas de tiempo pueden utilizarse de forma:

- **Estática**: se requiere conocer previamente el proceso para establecer un orden en las marcas de tiempo
- **Dinámica**: las marcas de tiempo son asignadas conforme las transacciones son recibidas

Para ambos casos se establecen las reglas que deben seguir las transacciones del proceso. En caso de que no se lleguen a cumplir, la transacción que generó el conflicto se aborta y se revierte su estado, otorgando a la transacción una nueva marca de tiempo para que vuelva a ejecutarse en un tiempo posterior. El problema que tiene esta técnica es que puede ocasionar una espera indefinida de la ejecución de la operación.

Técnicas multiversión

Esta técnica guarda varias versiones del valor que tienen los datos que se ha actualizado y permite la ejecución de las operaciones de lectura y escritura del dato. La lectura toma la última versión disponible del dato, mientras que la escritura genera una nueva versión del dato. Esta técnica requiere mucha capacidad de almacenamiento de información.

Protocolos optimistas de validación

Esta técnica no realiza ninguna verificación durante la ejecución de una operación; cuenta con cuatro transacciones: lectura de datos, ejecución, validación y escritura de datos. El sistema lee el dato a modificar y realiza una copia local sobre la cual se ejecutan los cambios. Al terminar de ejecutar la operación se realiza una validación que comprueba que no se han ocasionado conflictos en los datos para finalmente escribir el dato en el sistema.

Técnicas de bloqueo

Esta técnica controla el acceso concurrente de las transacciones y se basa en el concepto de bloquear los recursos. La técnica de bloqueo sigue las siguientes reglas:

- Se puede emitir una señal de bloqueo antes de ejecutar una operación sobre un recurso compartido
- Se puede emitir una señal de desbloqueo al terminar una operación sobre un recurso compartido
- No se puede emitir una señal de bloqueo sobre un recurso compartido previamente bloqueado
- No se puede emitir una señal de desbloqueo sobre un recurso compartido a menos que sea el proceso que posee el uso del recurso.

La técnica de bloqueo utiliza un tipo de granularidad[1], en la cual se realiza una selección del número de elementos que componen un objeto a bloquear. El tamaño del objeto depende del tipo de transacción que se ejecutará sobre ellos; es una buena técnica ya que permite un nivel de granularidad múltiple.

Un problema que se presenta en esta técnica es el interbloqueo, el cual ocurre cuando una operación bloquea el recurso y espera una instrucción para liberarlo, esto puede provocar que los procesos esperen indefinidamente la transacción de liberación del recurso y que dicho recurso se mantenga bloqueado.

Para el proceso de control de concurrencia, se optó por incluir en *Misho* el control de concurrencia a base de bloqueos por granularidad. La granularidad depende de los objetos que maneja el espacio de producción, i.e. los objetos que manejan el editor de texto y el pizarrón respectivamente (se explicará la granularidad utilizada posteriormente, en la sección 3.4).

[1]Se denomina granularidad al tamaño de los elementos pertenecientes a un objeto

3.3.3. Lenguajes de programación

El desarrollo de aplicaciones de software para las computadoras de escritorio depende principalmente del sistema operativo que esté instalado en éstas. Existen tres sistemas operativos de escritorio principalmente: Linux, Mac OSX y Windows. Debido a la poca compatibilidad que existe entre estos sistemas operativos (principalmente entre Windows y los demás) pocas aplicaciones pueden ser ejecutadas en las tres plataformas.

Dentro de los lenguajes de programación de aplicaciones de escritorio que se analizaron para el desarrollo de *Misho* están:

Java: es un lenguaje de programación orientado a objetos, altamente portable debido a que las aplicaciones desarrolladas en Java están compiladas en archivos byte code, mismos que interpretan la máquina virtual de Java por cada sistema operativo, lo cual lo vuelve independiente a la plataforma.

Visual Basic: es un lenguaje de programación que permite el desarrollo de aplicaciones de forma simple; fue desarrollado por Microsoft por lo que las aplicaciones de Visual Basic sólo puede ser ejecutados en Windows.

C #: es un lenguaje de programación orientado a objetos y a componentes, que pretende incorporar las ventajas de programación que tiene los lenguajes Java y C++ en el desarrollo de aplicaciones. Originalmente C # se creó para desarrollar programas sobre la plataforma .NET[1], la cual limita sus aplicaciones a ser ejecutadas en Windows, posteriormente se estandarizaron las especificaciones del lenguaje y la plataforma para que sus aplicaciones puedan ser ejecutadas en otros sistemas operativos utilizando una versión libre (Mono Hispano[2]) .

Objective-C: es un lenguaje de programación orientado a objetos similar a C++; utilizado por Apple para el desarrollo de las aplicaciones para Mac OSX e iOS.

Haciendo una comparación de los cuatro lenguajes descritos anteriormente, se optó por utilizar el lenguaje de programación Java, el cual permite desarrollar aplicaciones multiplataforma.

3.3.4. Tecnologías de comunicación en red

Dado que se optó por elegir Java como lenguaje de programación para la implementación de *Misho*, se estudiaron las tecnologías que permiten la comunicación en red, encontrando tres opciones:

[1] La plataforma .NET es una interfaz de programación para aplicaciones montada sobre Windows
[2] http://monohispano.org/ecma/

Biblioteca *JXTA*: es una plataforma de red abierta que provee los bloques básicos de construcción de un sistema *peer-to-peer*.

Sockets: permite crear una conexión *cliente-servidor* utilizando un mecanismo de intercambio de mensajes vía TCP

Biblioteca *KryoNet*: permite crear una conexión *cliente-servidor* utilizando un mecanismo de intercambio de mensajes vía TCP y UDP

Se realizó una comparación de las tres tecnologías, con la que se obtuvo el siguiente resultado: "La biblioteca *JXTA* es complicada de utilizar, debido a la gran cantidad de información redundante que se envía a través de la red ademas se tendría que adaptar la técnica de control de concurrencia a un protocolo optimista de validación para evitar inconsistencias en la edición del documento o la imagen; mientras que la utilización de sockets requiere la serialización de los objetos que se envían por la red y el tiempo de respuesta es considerable en caso de querer realizar una aplicación en tiempo real. Por estos motivos ambas tecnologías fueron rechazadas.

La biblioteca *KryoNet* es utilizada como soporte de comunicación en videojuegos sobre Internet, lo cual implica que el tiempo de respuesta es aceptable para una aplicación en tiempo real. Además utiliza una biblioteca de serialización de lectura y escritura de objetos llamada *Kryo* (Kryo serializa los objetos, i.e. los vuelve "archivos binarios"que pueden enviarse por la red como cualquier otro archivo, y reconstruirse del otro lado) mediante la cual es simple poder transmitir la información en una arquitectura *cliente-servidor*. También *KryoNet* puede ser utilizada en aplicaciones de escritorio y para dispositivos móviles con plataforma *Android*, por lo cual se optó por utilizar esta biblioteca."

Existen dos tipos de tecnologías que se refieren al origen donde se realiza la petición de una transacción en una arquitectura cliente-servidor. Estas son:

Tecnología *push*: la petición de una transacción es originada desde el servidor

Tecnología *pull*: la petición de una transacción es originada desde el cliente.

Se optó por utilizar la tecnología *push*, para tener un control en las peticiones de las transacciones desde el servidor y así disminuir el procesamiento de información que realiza cada cliente.

Al utiliza una red de computadoras se requiere establecer el protocolo de comunicación para la transmisión de mensajes. Los dos protocolos principales de Internet se describen a continuación:

TCP (*Transmission Control Protocol*): proporciona una entrega fiable y ordenada de una serie de paquetes de datos enviada de una computadora a otra

UDP (*User Datagram Protocol*): envía los mensajes mediante datagramas a otros hosts en la red, sin que se haya acordado una comunicación previa que establezca los canales de transporte o rutas de los datos. Además que no asegura el orden de los paquetes de datos.

Debido a la confiabilidad que presenta el protocolo TCP en la entrega de paquetes de datos, se optó por utilizar este para la comunicación en red.

3.4. Análisis de las funcionalidades de *Misho*

Misho, el entorno cooperativo propuesto en esta tesis de maestría, es una aplicación *Groupware* enfocada en dar solución a los problemas que enfrentan los colaboradores al editar cooperativamente un documento, mismos que se describieron en la sección 3.1 de este capítulo (página 49).

Dado que los colaboradores pueden estar distribuidos en tiempo y espacio, *Misho* debe estar basado en una arquitectura distribuida, i.e. esta aplicación utiliza una red de computadoras para el intercambio de información.

Misho busca ofrecer una conciencia de trabajo en grupo, con la integración de los tres espacios funcionales de *Groupware*: *producción*, *comunicación* y *coordinación*. A continuación, se describen los objetivos que tiene *Misho* al incluir estos espacios.

3.4.1. Espacio de producción

Misho utiliza **espacio de producción** para generar un ambiente cooperativo donde los colaboradores realicen la edición de documentos HTML e imágenes vectoriales.

Para facilitar la utilización del entorno en cada una de las actividades que realizan los colaboradores, se desarrollaron dos módulos independientes que son integrados en la interfaz gráfica de *Misho*. El primer módulo es un editor cooperativo de texto, el cual permite la creación y edición de los documentos HTML; y el segundo módulo es un pizarrón cooperativo, el cual permite la creación y edición de imágenes vectoriales.

Misho está construido sobre un diseño orientado a objetos, que establece una estructura lógica y jerárquica que permite almacenar, organizar, manipular y acceder a la información de los documentos o imágenes respectivamente. A continuación, se describirá el modelo de datos dependiendo de los requerimientos funcionales de cada módulo que comprende el espacio de producción.

Editor cooperativo de texto

Lo que se buscó al ofrecer un editor cooperativo de texto es que los colaboradores tengan un espacio donde puedan crear, organizar, visualizar, editar y aprobar un documento de forma conjunta.

Se optó por utilizar el formato HTML para exportar la información del documento debido a que permite al colaborador establecer un estilo en la fuente del texto del documento, pero el modelo de datos utilizado para el almacenamiento de la información no se encuentra ligado a un formato en específico, i.e. el modelo de datos de la información del documento contiene banderas que indican el estilo que debe aplicarse al dato con el fin de facilitar su exportación al formato HTML.

Como se mencionó en el capitulo introducción (página 7) al trabajar con documentos muy extensos se requiere ofrecer a los colaboradores dos cosas: 1) tener una organización jerárquica de la información para mantener un orden en el documento y 2) limitar el acceso a secciones del documento con la finalidad que los colaboradores puedan acceder a las secciones del texto de su interés. Por estos motivos se optó por fragmentar la información de un documento en secciones y así proporcionar al grupo de trabajo la opción de que cada colaborador trabaje en una sección distinta del documento.

Otra de las características, que debe ofrecer el editor cooperativo de texto es la conciencia de que se esta trabajando en grupo, para ello se requiere que el editor integre un proceso que notifique a cada colaborador los cambios realizados por los demás colaboradores. Sin embargo, como se explicó en la sección anterior, esto depende del rol que desempeña cada colaborador, por lo que se optó en ofrecer la posibilidad de que el colaborador elija entre: 1) una sincronización síncrona, compartiendo y recibiendo cada acción realizada en el editor de texto que afecte al documento; 2) una sincronización asíncrona, almacenando los cambios localmente y después solicitar compartir y actualizar la información; y finalmente 3) una sincronización por número de transacciones donde la granularidad de sincronización que puede escoger es: una letra, una palabra o un párrafo.

También es necesario ofrecer a los colaboradores mantener una coherencia en la información del documento. Como se mencionó en la sección anterior (página 58) se utilizó la técnica a base bloqueos por granularidad como una forma de control de concurrencia, i.e. utilizar candados para limitar la edición de la información a un colaborador a la vez. Para ello, es importante definir cuál será la granularidad mínima a la que se le aplicará el candado considerando los elementos que componen la información del documento.

Primero se pensó en que el elemento más pequeño en la información es un caracter, sin embargo, la información agrupa un número de caracteres en un objeto denominado *palabra*. Como se vió anteriormente, en el estudio de los editores cooperativos existentes, el tener una granularidad por letra pertmite que dos personas escriban en una misma

palabra y se ocasione un problema de sintaxis en dicha palabra, por lo que se optó que *Misho* utilizará como granularidad mínima "palabra", con el fin de asegurar la coherencia en la información. sin embargo, si se aplica el candado solamente a la palabra, *Misho* no puede asegurar un verdadero control en el contenido de la información del documento, para evitar este problema, se requirió utilizar el candado sobre un conjunto de tres palabras en el proceso de edición, las palabras a las que se aplica el candado son: 1) la palabra que está editando cada colaborador, 2) la palabra que precede a la palabra en edición y 3) la palabra que antecede a la palabra de edición. El candado se quitará una vez que el colaborador haya terminado de editar la sección de texto del documento.

Para poder informar a los colaboradores que un conjunto de palabras esta siendo utilizado por algún colaborador debe existir una notificación en el editor de texto, pero esta notificación no debe afectar a las actividades que realiza cada colaborador, por lo que se optó por utilizar un resaltado de texto al cambiar el color del texto a un tono gris, con la opción de que al dar click izquierdo con el ratón sobre la sección de texto se notifique que colaborador es el que se encuentra editando dicha sección.

Otro proceso que debe tomarse en consideración en el editor de texto es como puede ser expresada la *deixis* hecha desde el espacio de comunicación, recordando que el proceso de *deixis* es un señalamiento a una sección de texto dentro del documento. Para este proceso se consideró desplegar al usuario la sección del documento que contiene dicho señalamiento así como resaltar el texto cambiando el fondo de la sección del texto a un color cyan, con la finalidad de que pueda ser fácilmente localizado por los colaboradores. La sección del texto vuelve a su estado original cuando el colaborador realiza una acción sobre la sección en el editor de texto.

Finalmente, el proceso de aprobación de la información requiere un consenso que verifique que todos los colaboradores, cuyo rol es de *aprobador*, han aceptado dicha versión de la sección y notificar a todos los colaboradores para evitar que se sigan realizando modificaciones en la información de dicha sección.

Pizarrón cooperativo

Lo que se buscó al ofrecer un pizarrón cooperativo es que los colaboradores tengan un espacio donde puedan crear, organizar, visualizar, editar y aprobar una imagen de forma conjunta.

Se optó por la edición de imágenes vectoriales, debido a que una imagen vectorial es una imagen digital formada por objetos geométricos independientes (líneas, círculos, polígonos, etc.), a los cuales se les denomino *figura* en *Misho*. Cada figura es definida por distintos atributos, como son: forma, de posición, de color, etc.

Misho ofrece a los colaboradores la opción de utilizar las figuras básicas convencionales, i.e. el colaborador puede insertar en la imagen figuras de tipo: línea recta, cuadrado, círculo y triángulo. Como una imagen también puede contener texto se incluyó esta posibilidad.

El pizarrón cooperativo además de permitir insertar figuras, requiere proporcionar la opción de modificar las propiedades de dichas figuras. Las propiedades que se consideraron permitir modificar en las figuras de la imagen son: posición de la figura, tamaño de la figura, tamaño del borde, color del borde y color del relleno.

Como se puede observar la granularidad mínima de transacción en el pizarrón cooperativo son las propiedades de la figura, pero no es práctico utilizar esta granularidad, debido a que en realidad afectaría al elemento *figura* por tal motivo, se consideró la granularidad de figura para los procesos de candados y sincronización.

La sincronización que puede ser ofrecida por *Misho* en el módulo del pizarrón cooperativo es síncrona o asíncrona. En la sincronización síncrona, los cambios realizados en una figura de la imagen son compartidos y actualizados en el lienzo de cada colaborador, mientras que en la sincronización asíncrona, el pizarrón cooperativo guarda localmente los cambios realizados por el colaborador en espera de que el colaborador realice la petición de compartir y actualizar información.

El proceso de control de concurrencia mediante el uso de candados en el pizarrón cooperativo se realiza a nivel figura, cuando un colaborador selecciona una imagen para editar sus propiedades, es necesario bloquear el acceso de edición a esta figura a los demás colaboradores, hasta que el usuario haya terminado de modificar sus propiedades. En este proceso es necesario notificar a los demás colaboradores que figuras están siendo utilizadas, sin que la notificación afecte las actividades que éste realiza. Por ello, se optó por resaltar la figura que esta bloqueada por otro colaborador, cambiando el color del borde y relleno de la figura a un color gris, para que cuando un colaborador realice un click izquierdo con el ratón se muestre el nombre el colaborador que esta editando dicha figura.

Otro proceso que debe tomarse en consideración en el pizarrón cooperativo es el de expresar la *deixis* realizada desde el espacio de comunicación, recordando que el proceso de deixis es un señalamiento a una figura dentro de la imagen. Para este proceso se consideró resaltar la figura señalada, cambiando el color del borde de dicha figura un color cyan de forma intermitente, i.e. alternando el color del borde de la figura del color original a un color cyan por periodos de un segundo. La figura vuelve a su estado original cuando un colaborador realiza una acción sobre el lienzo de la imagen.

Finalmente, el proceso de aprobación de la información requiere un consenso que verifique que todos los colaboradores, cuyo rol es de *aprobador*, han aceptado dicha versión de la sección y notificar a todos los colaboradores para evitar que se sigan realizando modificaciones en la información de dicha imagen.

3.4.2. Espacio de comunicación

El espacio de comunicación busca ofrecer a los colaboradores un medio sobre el cual puedan intercambiar ideas y discutir secciones del documento o de la imagen. Este espacio esta compuesto por una sala de mensajería instantánea que permite el intercambio de información a través del uso de mensajes de texto.

Misho ofrece comunicar al colaborador en cuestión con cada uno de los colaboradores que se encuentren en la misma sesión y una sala general para comunicarse con todos lo colaboradores a la vez.

A diferencia de otras salas de mensajería instantánea, este espacio permite realizar un proceso de *deixis* sobre los módulos que comprende el espacio de producción (editor cooperativo de texto y pizarrón cooperativo).

El proceso de deixis utiliza un conjunto de hipervínculos para hacer referencia a la información señalada en el espacio de producción. El hipervínculo consiste en crear una etiqueta dentro del mensaje a compartir, la cual contiene la ubicación de la información señalada en el espacio de producción, esta etiqueta se distingue por tener un formato diferente al texto del mensaje. Cuando este hipervínculo es compartido en la sala de mensajería instantánea, los colaboradores a los que se compartió el mensaje pueden dar click izquierdo sobre la etiqueta y en automático *Misho* señalará la información en el espacio de producción donde se generó este señalamiento.

3.4.3. Espacio de coordinación

El espacio de coordinación busca ofrecer mecanismos que faciliten a los colaboradores la asignación de tareas en la edición cooperativa y proporcionar un soporte en toma de decisiones del grupo de trabajo.

Como la finalidad de *Misho* es permitir la edición de documentos e imágenes por un grupo de trabajo, se tomó en consideración que en una organización pueden existir diferentes grupos de trabajo que realizan la edición de distintos documentos. Por tal motivo *Misho* organiza a las personas que lo integran por defecto en un grupo *publico* y permite la creación de grupos de trabajo independientes donde los colaboradores se integran mediante invitación a éste.

Misho permite a cada grupo de trabajo organizar los documentos y las imágenes que son editadas, revisadas y aprobadas por los colaboradores de dicho grupo. También permite a los colaboradores establecer el rol que tiene cada colaborador en la edición de las secciones de los documentos o imágenes.

Otra funcionalidad que proporciona el espacio de coordinación es la toma de decisiones utilizando un proceso de consenso entre los colaboradores que tienen el rol de *aprobador* para decidir si la información de la sección del documento o la imagen (respectivamente) se encuentra aprobada. Una vez aprobada la información, *Misho* notifica a todos los colaboradores que la versión actual de la sección del documento o la imagen es la versión final y no requiere más modificaciones.

Finalmente el espacio de coordinación es el encargado de administrar las actividades que realizan todos colaboradores y la información generada por los mismos.

3.5. Diseño de *Misho*

El diseño de *Misho* se basa en una arquitectura cliente-servidor, donde el servidor es el encargado de procesar y administrar los cambios globales de cada grupo de trabajo, mientras que el cliente procesa localmente los cambios realizados por el colaborador que lo utiliza.

El cliente es el entorno cooperativo que integra los tres espacios funcionales de *Groupware* mediante el cual interactúa cada colaborador del grupo de trabajo, mientras que el servidor es un proceso que controla y administra las actividades realizadas por estos colaboradores en dicho entorno.

En esta sección se describen las consideraciones que se tomaron en cuenta en el diseño de *Misho*.

3.5.1. Diseño del servidor de *Misho*

El servidor centraliza toda la información generada en *Misho*, tiene como finalidad administrar, procesar y compartir dicha información entre los participantes que lo integran. Además, el servidor tiene la función de administrar: 1) los usuarios del sistema (colaboradores), 2) los grupos de trabajo, 3) los documentos o imágenes de cada grupo de trabajo, y finalmente 4) los permisos que tiene cada usuario sobre dichos documentos o imágenes.

Las acciones realizadas por los usuarios presenten en la sesión, generan peticiones que son enviadas al servidor para ser procesadas y enviar como respuesta una notificación a todos usuarios que se encuentran involucrados en dicha petición, lo que representa la actualización de la información en todos los clientes conectados de forma síncrona. En este proceso, el servidor almacena la información de dicha petición y mantiene una única versión de la sección del documento o la imagen. Los usuarios puede acceder a dicha versión y copiarla para ser modificada localmente.

El servidor mantiene un registro de los elementos que han sido cambiados durante una sesión y además un registro de los usuarios que cuentan con la última versión de la información del documento o la imagen, se diseñó de esta forma de con la finalidad de que si un usuario se conecta o solicita la actualización de la información en el espacio de producción sólo sean enviados los cambios que él no conoce aún.

Cada usuario puede elegir el tipo de sincronización para actualizar la versión de cada sección del documento o imagen que utiliza: síncrona o asíncrona. Cuando un usuario utiliza una sincronización síncrona, las acciones que éste realiza sobre el entorno *Misho*, son enviadas al servidor para ser almacenadas, procesadas y retransmitidas a todos los usuarios con los que está trabajando en conjunto, como una notificación de un cambio realizado en la información. Por el contrario, los usuarios que eligieron una sincronización asíncrona reciben bajo petición.

En resumen, el diseño de *Misho* se basa en un flujo de peticiones y respuestas entre el usuario C_x, *Misho EC* y el servidor S. El flujo es el siguiente: 1) el usuario realiza una acción sobre *Misho*, 2) *Misho* procesa la información y envía una petición al servidor (en caso de ser requerido), 3) el servidor recibe la petición y procesa la información enviando como respuesta una notificación de la información actualizada a *Misho*, 4) *Misho* procesa la respuesta y despliega la información al usuario. Un ejemplo de este proceso ocurre cuando un usuario solicita su registro a *Misho*, el flujo de peticiones y respuestas presentes entre el usuario, *Misho* y el servidor se muestra en la figura 3.8.

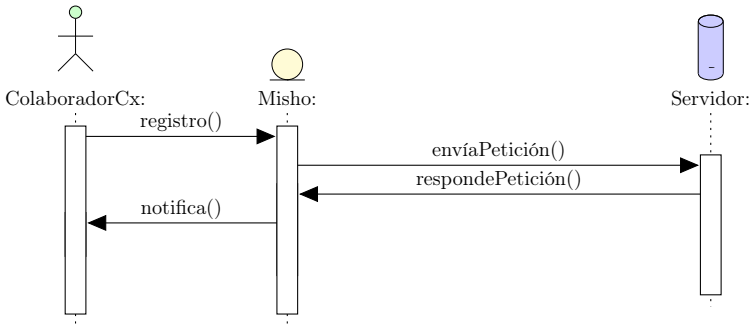


Figura 3.8: Ejemplo del flujo de información entre el usuario, *Misho* y el servidor al registrar un nuevo usuario

3.5.2. Diseño del cliente (entorno) de *Misho*

Misho requiere el diseño de una interfaz gráfica que incluya los módulos que comprenden los espacios de producción y comunicación en una misma ventana, i.e. la ventana

de *Misho* está dividida en tres secciones que contienen el editor cooperativo de texto, el pizarrón cooperativo y la sala de mensajería instantánea. Se optó por ofrecer al usuario la capacidad de ocultar los módulos que éste no este utilizando en un momento dado.

Como toda aplicación, *Misho* cuenta con una barra de menús en la parte superior de la ventana, misma que contiene los menús concernientes al grupo de trabajo (menú *Workteam*), editor cooperativo de texto (menú *Editor*) y pizarrón cooperativo (menú *Blackboard*).

Menú **Grupo de trabajo (*Workteam*)**: contiene la configuración básica los grupos de trabajo, está conformado por:

- Grupo de trabajo (*Workteam*): permite crear de un nuevo grupo de trabajo, así como realizar invitaciones a los colaboradores de la organización para participar en dicho grupo. También ofrece la opción de eliminar a los colaboradores de los grupos de trabajo.

- Registrar usuario (*Sign Up*): permite registrar a un colaborador como usuario de *Misho*

Para el diseño de la interfaz gráfica del editor cooperativo de texto, se tomó en consideración el estándar impuesto por los editores de texto más comunes (Microsoft Office y OpenOffice), cuya tendencia es mantener la barra de herramientas en la parte superior, misma que contiene íconos con acceso directo a las funcionalidades de la aplicación. De igual forma, se optó porque la interfaz pudiera manejar varios documentos a la vez y puedan ser visualizados a través de pestañas en la parte inferior de la ventana.

Cabe mencionar que para hacer más fácil al usuario el manejo de un documento, se creó una estructura del mismo a base de secciones, i.e. un documento contiene un número limitado de secciones establecidas por el usuario. Para poder visualizar estas secciones se pensó en el mismo mecanismo de visualización a través de pestañas que se encuentren en el lado izquierdo del editor de texto.

Los requerimientos funcionales que debe cumplir el editor cooperativo de texto (los cuales están incluidos en el menú de *Misho* llamado *Editor*) son:

- **Menú Archivo (*File*)**: este menú contiene la configuración básica de un documento.

  - Nuevo (*New*): permite la creación de un nuevo documento[1]
  - Abrir (*Open*): permite abrir un documento creado anteriormente en *Misho*.

[1] Al crear un nuevo documento se genera automáticamente una sección con el mismo nombre donde el usuario puede empezar a trabajar.

- Guardar (*Save*): permite guardar los cambios efectuados en la sección del documento que esté activa.
- Guardar como (*Save As*): permite realizar una copia de la sección del documento con un nuevo nombre, en caso de que ésta ya esté guardada en el servidor.
- Exportar (*Export*): permite exportar el texto a un formato HTML o PDF.

- **Menú Actualización (*Update*)**: este es un menú que maneja las opciones mediante las cuales se puede actualizar la sección del documento.
  - **Sub-menú Sincronización (*Synchronization*)**: contiene el sub-menú que permite establecer el tiempo en el cual se actualiza la información en el servidor, las opciones que presenta este sub-menú son excluyentes entre sí.
    - Síncrona (*Synchronous*): los cambios se actualizan automáticamente cuando se cumple el numero de transacciones indicadas por la granularidad .
    - Asíncrona (*Asynchronous*): a petición del usuario, la información se actualiza en el servidor y es compartidos a los demás usuarios.
  - **Sub-menú Granularidad (*Granularity*)**: contiene el sub-menú que permite establecer la granularidad de texto a la cual se va actualizar la información en el servidor.
    - Caracter (*Letter*): cada vez que el usuario ingrese un carácter, éste será enviado al servidor.
    - Palabra (*Word*): cada vez que el usuario ingrese una palabra, cuyo fin se indica por un carácter de tipo espacio, la información será enviada al servidor.
    - Párrafo (*Paragraph*): cada vez que el usuario ingrese un salto de línea, la información será enviada al servidor.
- **Menú Edición (*Edition*)**: contiene el menú de opciones que permite la edición de la fuente utilizadas en el documento.
  - Tipo de Fuente (*Type Font*): permite establecer el tipo de fuente de la palabra en la sección del documento actual. Las opciones que presenta son: ***Arial***, ***Comic Sans***, ***Impact***, ***Monospace***, ***Times New Roman***.
  - Tamaño de Fuente (*Size Font*): permite establecer el tamaño de la fuente de la palabra en la sección del documento actual. Los tamaños varían desde 8 hasta 28 pixeles.
  - Estilos de la Fuente (*Styles*): permite modificar el estilo de la fuente de la palabra seleccionada en la sección del documento actual. Los estilos a escoger son: ***Negrita*** A, Cursiva A, ***Subrayado*** $\underline{A}$, ***Subíndice*** A_x, ***Superíndice*** A^x.
  - Alineación de párrafo (*Alignment*): permite modificar la alineación del párrafo en la sección del documento actual. La alineación puede ser: ***Izquierda***, ***Derecha***, ***Centrada*** y ***Justificada***.

- **Menú Insertar (*Insert*)**: ofrece la opción de insertar una imagen en el documento.
  - Insertar (*Insert*): inserta una imagen en la sección del documento actual.
  - Alineación de imagen (*Alignment*): permite alinear la imagen con respecto al párrafo en la sección del documento. La alineación puede ser: ***Izquierda***, ***Derecha***, ***Centrada*** y ***Justificada***.
- **Menú Herramientas (*Tools*)**: brinda opciones sobre actividades que se pueden hacer en la sección del documento actual.
  - Compartir (*Shared*): permite compartir el texto seleccionado en la sección del documento actual. Este proceso es incluido en el proceso de *deixis*, que será descrito en el espacio de comunicación.
  - Aprobar (*Approve*): permite aprobar la sección del documento actual, siempre y cuando el usuario cuente con el permiso correspondiente.
- **Barra de Herramientas**: contiene los accesos directos a los menús de edición y herramientas.
- **Área de Trabajo**: donde se puedan incluir caracteres alfanuméricos y símbolos.
- **Lista de Documentos**: presenta una lista de documentos abiertos a los cuales tiene permiso el usuario.
- **Lista de Secciones**: presenta una lista de las secciones del documento a las cuales tiene permiso el usuario.

El modelo de datos es la representación jerárquica que ordena los objetos de las clases que conforman un documento. A continuación se presenta el diagrama de clases que constituye al modelo de datos de editor de texto (véase figura 3.9).

Para el diseño de la interfaz gráfica del pizarrón cooperativo, se tomaron las mismas consideraciones del editor cooperativo de texto, i.e. se diseñó una barra de herramientas en la parte superior, la cual contiene íconos con acceso directo a las funcionalidades de la aplicación; así como el uso de pestañas para la visualización de imágenes del pizarrón en la parte inferior de la ventana.

Los requerimientos funcionales que debe cumplir el pizarrón cooperativo (los cuales son incluidos en el menú de *Misho* llamado *Blackboard*) son:

- **Menú Archivo (*File*)**: este menú contiene la configuración básica de un documento.
  - Nuevo (*New*): permite la creación de una nueva imagen.

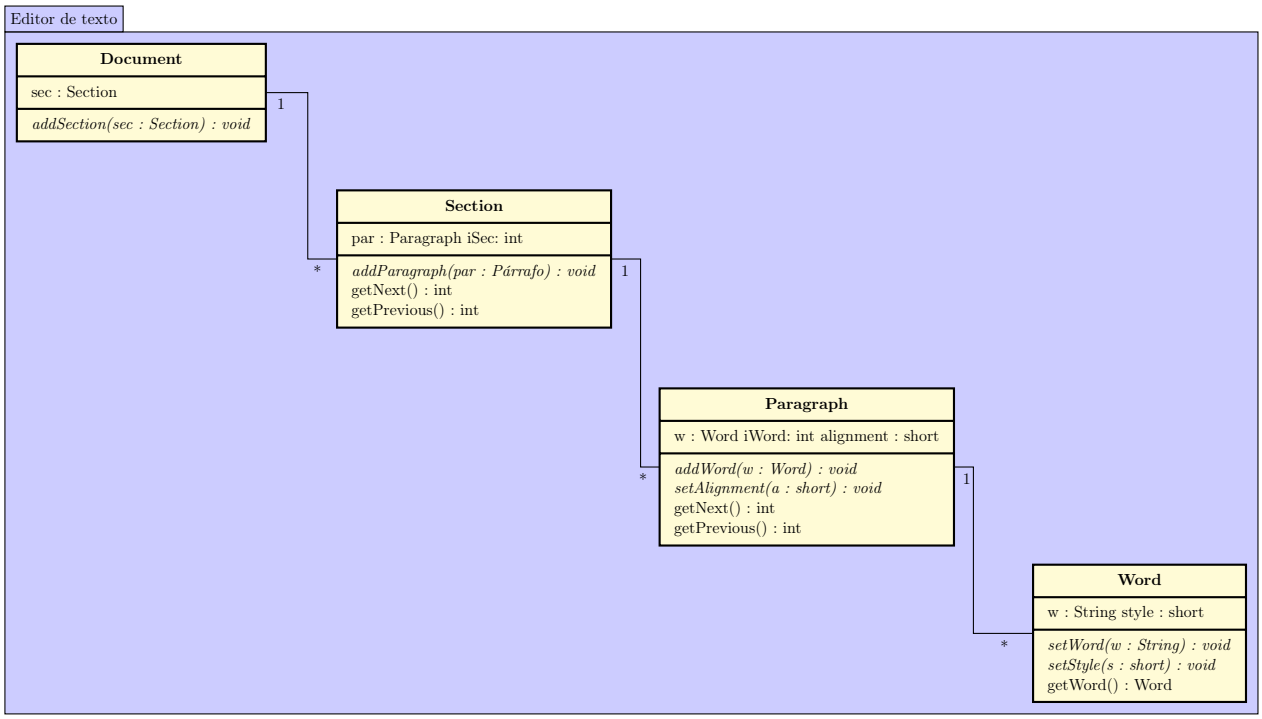


Figura 3.9: Modelo de datos del editor de texto

- Abrir (*Open*): permite abrir una imagen creada anteriormente en *Misho*.
- Guardar (*Save*): permite guardar los cambios en la imagen activa.
- Guardar como (*Save As*): permite realizar una copia de la imagen con un nuevo nombre, en caso de que ésta ya éste guardada en el servidor.
- Exportar (*Export*): permite exportar la imagen actual a un formato JPG y PNG.

- **Menú Sincronización (*Synchronization*)**: contiene las opciones que permite establecer el tiempo en el cual se actualiza la información en el servidor, las opciones que presenta este sub-menú son excluyentes entre sí.

  - Síncrona (*Synchronous*): se actualiza, cuando el usuario agregue o modifique una figura, enviando los cambios al servidor para poder ser compartidos a los demás colaboradores.
  - Asíncrona (*Asynchronous*): a petición del usuario, la información se actualiza en el servidor y es compartida a los demás colaboradores.

- **Menú Edición (*Edition*)**: contiene el menú de opciones que permite la edición de la imagen actual.

  - Agregar figura (*Add Figure*): permite agregar una figura a la imagen. Las figuras que el usuario puede seleccionar son: ***Texto***, ***Línea recta***, ***Círculo***, ***Cuadrado*** y ***Triángulo***.
  - Estilo de la Figura (*Styles*): permite modificar el estilo de las figuras de la imagen actual. Los estilos a escoger son: ***Cambiar tamaño del borde*** (*Size*), ***Cambiar color del borde*** (*Border*) y ***Cambiar el color del relleno*** (*Fill*).

- **Menú Herramientas (*Tools*)**: brinda opciones sobre actividades que se pueden hacer en la imagen actual.
  - Compartir (*Shared*): permite compartir la figura seleccionada en la imagen actual. Este proceso es incluido en el proceso de *deixis* que es descrito en el espacio de comunicación y producción.
  - Aprobar (*Approve*): permite al usuario aprobar la imagen, siempre y cuando tenga el rol de *aprobador*.
- **Barra de Herramientas**: la cual contiene los accesos directos a los menús de edición y herramientas.
- **Lienzo**: donde puedan agregar las figuras.
- **Lista de imágenes**: presenta una lista de imágenes abiertas, a las cuales tiene permiso el usuario.

El diagrama de clases que constituye al modelo de datos del pizarrón se presenta en la figura 3.10.

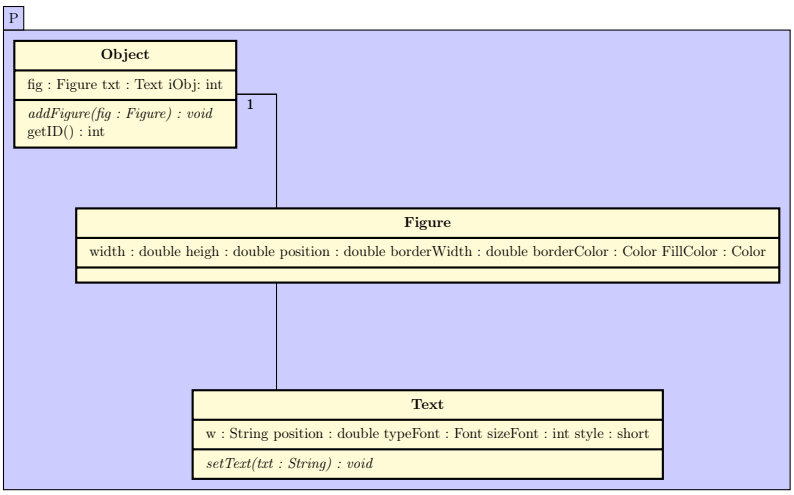


Figura 3.10: Modelo de datos del pizarrón

Para el diseño de la interfaz gráfica de la sala de mensajería instantánea se tomaron en consideración las características de las salas más comunes, tales como son Windows Live Messenger, gTalk, etc. que cuentan con una lista de usuarios conectados y una pestaña por cada conversación que se tiene establecida con los usuarios.

Los requerimientos funcionales de la sala de mensajería instantánea (los cuales son incluidos en el menú de *Misho* llamado *Chat*) son:

- **Área de mensajes**: espacio en el que son visualizados los mensajes de los colaboradores
- **Área de escritura de mensaje**: espacio en el que cada colaborador puede transmitir sus ideas a los demás colaboradores
- **Lista de colaboradores**: presenta una lista de los colaboradores conectados con los cuales se puede establecer una comunicación vía mensaje de texto.
- **Enviar mensaje (*Send*)**: envía el mensaje que ha escrito el usuario al servidor para ser compartido con el colaborador de la pestaña seleccionada
- **Proceso de *deixis***: se refiere al proceso de señalamiento que se realiza desde la sala de mensajería instantánea hacia los espacios de producción (editor cooperativo de texto y pizarrón cooperativo), i.e. que el hipervínculo en el área de mensajes el cual al ser seleccionado, resalta el texto o la figura compartida

El diagrama de clases que constituye al modelo de datos de la sala de mensajería instantánea se presenta en la figura 3.11.

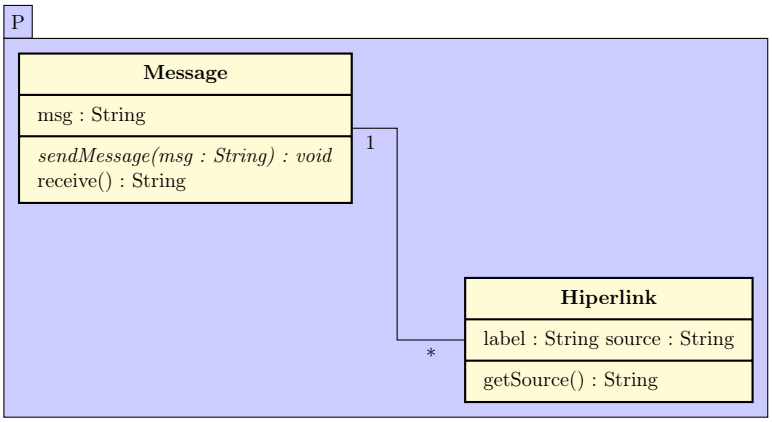


Figura 3.11: Modelo de datos de la sala de mensajería instantánea

Capítulo 4
Implementación

En este capítulo se presenta la implementación del diseño de *Misho*. Este capítulo se divide en tres secciones: la sección 4.1 describe como se implementó la arquitectura cliente-servidor de *Misho*, utilizando el patrón de desarrollo Modelo-Vista-Controlados; la sección 4.2: describe la implementación del cliente de *Misho*y la sección 4.3: describe la implementación del servidor de *Misho*.

4.1. Arquitectura de *Misho*

Como se explicó en el capítulo anterior, *Misho* integra los tres espacios funcionales de *Groupware* (comunicación, producción y coordinación) en un solo entorno cooperativo que permite la edición de documentos e imágenes vectoriales. Para su implementación se siguió el patrón de desarrollo de software Modelo-Vista-Controlador (*MVC*) [Gamma, 1995].

El patrón *MVC* se basa en dividir la implementación de una aplicación en tres partes: 1) un modelo de datos, 2) una vista y 3) un controlador. El modelo de datos está diseñado para almacenar y ordenar la información que se maneja en la aplicación; la vista se refiere a la interfaz de usuario que establece el medio de interacción entre el usuario y la aplicación; y finalmente, el controlador es el medio de comunicación que recibe las acciones generadas por los usuarios en la vista para consultar o modificar el contenido del modelo de datos y posteriormente desplegarlo en dicha vista.

En *Misho*, el patrón *MVC* se implementó de la siguiente forma. El cliente contiene: 1) una vista, 2) un modelo de datos local y 3) un controlador local que se comunica con servidor, usando la biblioteca *KryoNet*. El servidor contiene: 1) un controlador global, que acepta y responde peticiones de los clientes a través de *KryoNet*, 2) un modelo de datos global, el cual centraliza la información generada en *Misho* y 3) una vista simple, que inicia o detiene el servidor y permite monitorear las actividades de los colaboradores.

Para explicar la implementación de *Misho*, siguiendo el patrón *MVC*, se presenta la figura 4.1, en la que observan tres clientes conectados al servidor (clientes A, B y C). El cliente cuenta con un controlador local que atiende las acciones que realiza cada usuario en su vista, misma que comprende al editor cooperativo de texto, al pizarrón y a la sala de mensajería instantánea. El cliente contiene un modelo de datos local con la información de los documentos e imágenes que el usuario ha solicitado en cada sesión.

El servidor cuenta con un controlador global que atiende las peticiones que realizan todos los usuarios a través de la red y modifica el contenido del modelo de datos global que contiene toda información que se genera en *Misho*. Para administrar y organizar los usuarios, los roles, los documentos, las imágenes y los permisos a éstos, el modelo de datos global del servidor está conformado por dos partes: 1) una base de datos SQLite y 2) una estructura jerárquica de objetos en Java con el contenido de los documentos e imágenes.

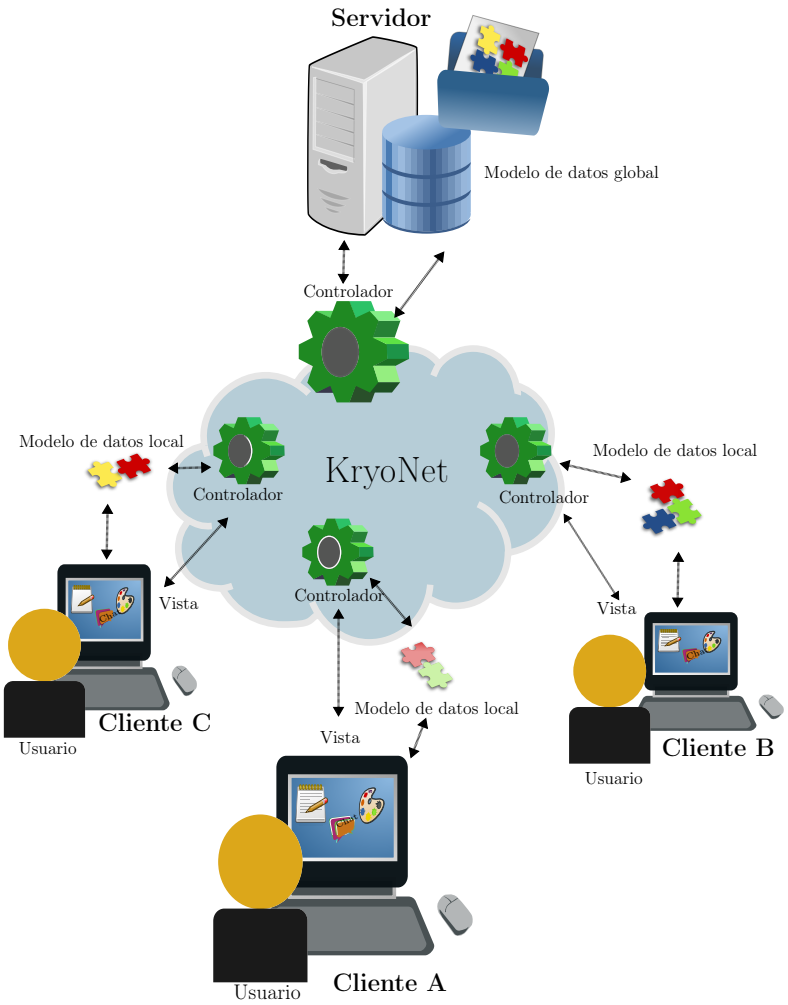


Figura 4.1: Modelo-Vista-Controlador de *Misho*

La comunicación entre el cliente y el servidor se realiza mediante la biblioteca *KryoNet*, a través de la cual se envían los objetos Java que contienen la información relacionada a las peticiones realizadas entre el cliente y el servidor. *KryoNet* utiliza el protocolo de comunicación TCP, para enviar los paquetes de datos en la red, mismos que son objetos serializados por la biblioteca *Kryo*[1]. El algoritmo que utiliza *Misho* en el envío y la recepción de peticiones entre el cliente y servidor se describe a continuación.

Algoritmo 1 Envío y recepción de peticiones entre el cliente y el servidor

Entrada: *Obj*=objeto de la petición serializado mediante *Kryo*
1: El emisor envía *Obj* al receptor a través de *KryoNet*
2: El receptor recibe *Obj* y la procedencia del mismo //Solo si es el servidor requiere la procedencia del objeto serializado.
3: **si** la procedencia se encuentra activa **entonces**
4: **si** *Obj* pertenece al conjunto de clases registradas para petición **entonces**
5: El controlador del receptor verifica la clase a la que pertenece y notifica al controlador pertinente la petición a realizar
6: El controlador del receptor procesa la petición
7: **si** existe una respuesta **entonces**
8: El controlador del receptor envía la respuesta al emisor utilizando este algoritmo
9: **fin si**
10: **fin si**
11: **fin si**

Cuando el cliente realiza una consulta a la base de datos, para administrar los permisos de acceso o la configuración de los mismos, se utilizan dos clases: *DBOperationRequest* y *DBOperationResponse* (Véase figura 4.2). Los objetos de estas clases se serializan utilizando *Kryo* y se envía a través de *KryoNet*. Como existen varias clases que acceden a la base de datos, se optó por generalizar las peticiones y respuestas en las clases antes mencionadas. Los atributos que tienen estas clases genéricas incluyen: 1) un *String* para indicar el nombre de la clase que realiza la petición, 2) un *String* indicando la acción a realizar y 3) un *String* para los parámetros de la acción de la petición o un conjunto de *Strings* para las respuestas.

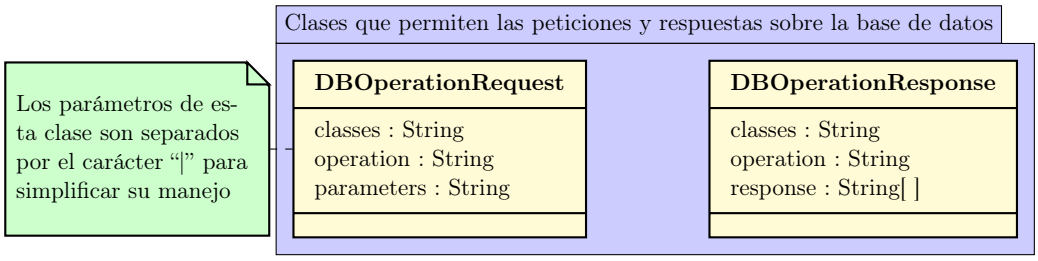


Figura 4.2: Clases que permiten las peticiones y respuestas sobre la base de datos

[1] *Kryo* convierte un objeto Java en un archivo binario y viceversa, con el fin de poder enviar objetos por la red y reconstruirlos al recibirlos

A continuación se explicará la implementación del cliente y servidor de *Misho*, siguiendo el patrón *MVC*.

4.2. Cliente de *Misho*

Como se observa en la figura 4.1 cada cliente está compuesto por una vista, un modelo de datos local y un controlador. La vista del cliente es la interfaz de usuario que incluye las ventanas del editor cooperativo de texto, el pizarrón cooperativo, la sala de mensajería instantánea y el conjunto de opciones que ofrece *Misho* para la administración de usuarios, documentos e imágenes.

El editor cooperativo de texto, el pizarrón cooperativo y la sala de mensajería instantánea constituyen objetos Java diseñados con el patrón *MVC*, motivo por el cual *Misho* ofrece la posibilidad de manejar diversas instancias de estos objetos, los cuales pueden ser accesibles a través de pestañas de la interfaz de usuario, e.g. la sala de mensajería instantánea permite una comunicación independiente entre pares de usuarios, donde cada conversación es un objeto Java y se accede a través de una pestaña de la sala de mensajería.

En la figura 4.3 se presenta la vista que agrupa los componentes de los espacios funcionales en *Misho*, la cual integra al editor cooperativo de texto, al pizarrón cooperativo, la sala de mensajería instantánea en una misma ventana y una barra de menús con las opciones de cada componente.

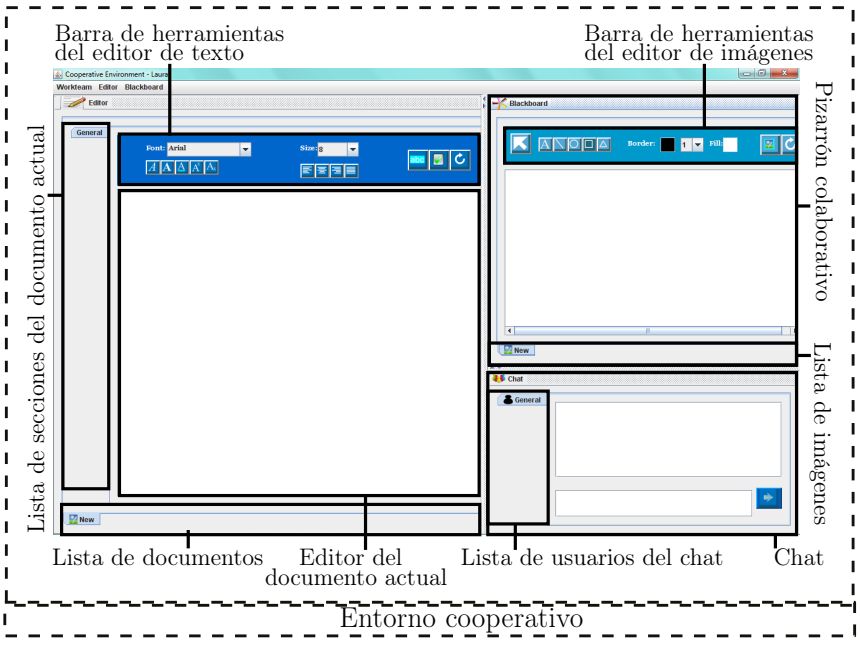


Figura 4.3: Interfaz gráfica de la aplicación *Misho*

A continuación, se describe la implementación de los objetos Java correspondientes al editor cooperativo de documentos, al pizarrón cooperativo y a la sala de mensajería.

4.2.1. Editor cooperativo de texto

El editor cooperativo de texto está compuesto por: 1) una vista que permite la creación, administración y edición de documentos; 2) un modelo de datos que agrupa jerárquicamente la información perteneciente a todos los documentos que el usuario está utilizando en cada sesión; y finalmente, 3) un controlador, que recibe y procesa las acciones que realiza el usuario sobre la vista para modificar o consultar el modelo de datos de forma local o global.

La vista del editor cooperativo de texto se muestra en la figura 4.4 y está compuesta por: 1) la lista de documentos (conjunto de pestañas) en la parte inferior de la ventana; 2) la lista de secciones pertenecientes a cada documento (conjunto de pestañas) en la parte izquierda de la ventana; 3) el panel de edición de la sección del documento, la cual contiene la barra de herramientas y el área de edición.

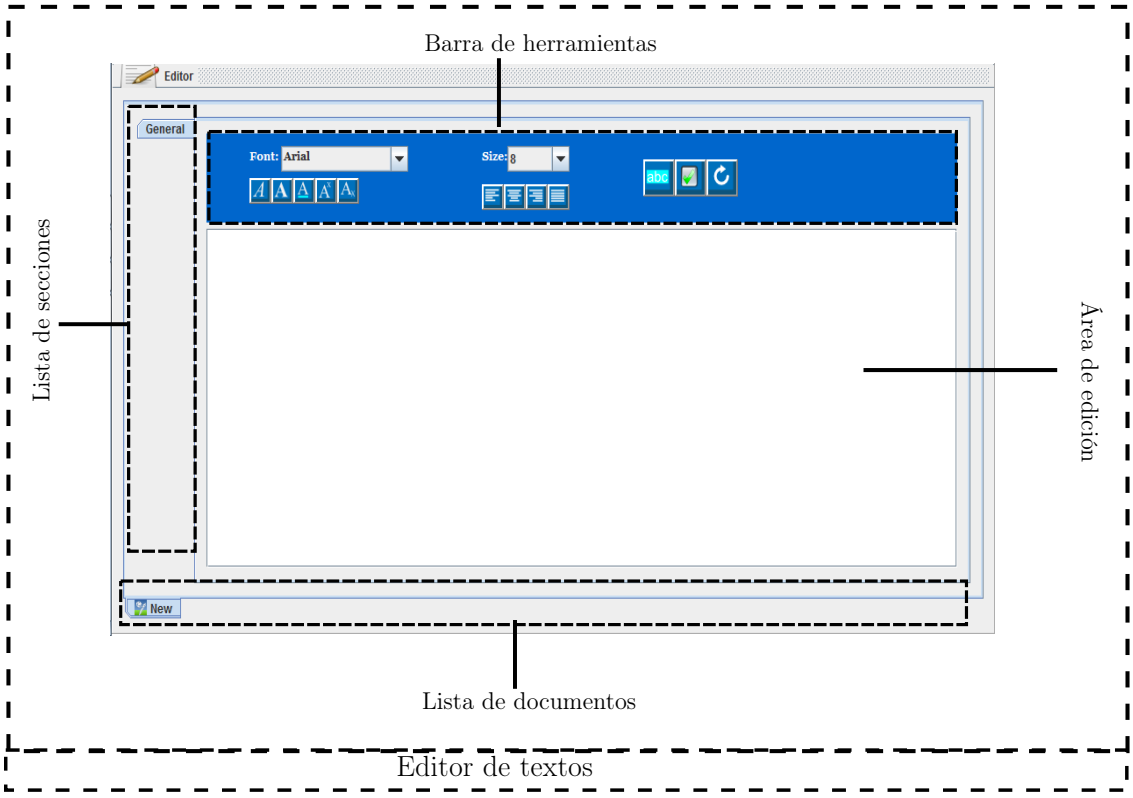


Figura 4.4: Interfaz de usuario del editor cooperativo de texto

Las acciones que puede realizar cada usuario sobre la vista del editor cooperativo de texto son:

1 *Crear un nuevo documento*
Esta acción crea un nuevo documento perteneciente a un grupo de trabajo específico. Automáticamente se genera una nueva sección en el nuevo documento, donde el usuario puede editar el texto de dicho documento.

2 *Abrir un documento existente*
Esta acción carga en pestañas las secciones en las cuales el usuario tiene permiso de visualización y su correspondiente el texto en cada sección.

3 *Seleccionar un documento*
Esta acción permite seleccionar un documento a través de una pestaña de la lista de documentos y a su vez, seleccionar una sección de la lista de secciones del documento activo. Los componentes sobre los cuales el usuario ejecuta acciones para la edición de la sección activa del documento son:

3.1 El área de edición para introducir o modificar el texto de la sección activa.

3.2 La barra de herramientas para modificar el estilo del texto de la sección. activa. Las opciones que ofrece la barra de herramientas son:

(a) Cambiar el tipo de fuente del texto seleccionado: Arial, Comic Sans, Impact, Monospace, Times New Roman.

(b) Cambiar el tamaño de la fuente del texto seleccionado: 8 a 28 px.

(c) Cambiar el estilo de la fuente del texto seleccionado: negrita, cursiva, subrayado, superíndice, subíndice.

(d) Cambiar la alineación del párrafo seleccionado: derecha, izquierda, centrada, justificada.

(e) Compartir un fragmento de texto de la sección del documento activa.

(f) Aprobar una sección del documento activa[1].

(g) Sincronizar la información de forma síncrona o asícrona con el servidor[2].

4 *Crear una sección en el documento activo*
Esta acción agrega una sección al documento activo. También permite invitar o anular la invitación a los usuarios pertenecientes al grupo de trabajo y establecer el rol que tiene cada usuario en la sección.

5 *Guardar el documento activo*
Esta acción guarda todos los cambios realizados en las secciones del documento.

[1] en caso de contar con el rol de *aprobador*
[2] en caso de tener seleccionada una sincronización asíncrona en la sección

6 *Cerrar el documento activo*
Esta acción cierra el documento activo. En caso de existir cambios, se notifica al usuario si se desean guardar.

7 *Exportar documento*
Esta acción permite exportar el documento (las secciones a las que tiene acceso el usuario) a un archivo con formato HTML y PDF.

El modelo de datos del editor cooperativo de texto se construyó de tal forma que cuenta con una lista de documentos, en la cual cada documento está dividido en una lista de secciones, donde cada sección está conformada por una lista se párrafos y, a su vez, cada párrafo está integrado por una lista de palabras. Esta organización permite manejar varios documentos de forma independiente.

Para organizar la información de cada elemento (sección, párrafo o palabra) se estableció una lista doblemente ligada para tener un seguimiento de cada elemento de la lista y facilitar las modificaciones de éstos al ser agregados o eliminados de la lista. Se utilizaron objetos de tipo *JTreeMap*[1] de Java, para almacenar la información, ya que dichos objetos agilizan la búsqueda de cada elemento, al utilizar sólo la clave que lo identifica.

La jerarquía de clases de un documento en el editor cooperativo de texto se presentan en la figura 4.5.

El controlador del editor cooperativo de texto controla tres aspectos que se trabajan conjuntamente, los cuales son: 1) recibir las peticiones del usuario a través de la vista, 2) manipular el contenido del modelo de datos local y 3) comunicar las acciones al servidor. Los métodos presentados en la figura 4.5 representan la parte del controlador que permite manipular la información de un documento.

Como se explicó en el capítulo anterior en la edición cooperativa de un documento se requieren funciones de: 1) edición de texto, 2) actualización de cambios, 3) administración de recursos compartidos, 4) modificación del estilo de la fuente del texto y la alineación del párrafo, y 5) soporte de *deixis*. A continuación, se explican los algoritmos que utiliza cada función anteriormente listada.

Edición de texto

El componente que permite editar y visualizar la información de cada sección es un objeto de tipo JEditorPane de Java, el cual maneja el texto como un arreglo de caracteres. En la figura 4.6 se ejemplifica el manejo del texto de una sección con respecto al componente JEditorPane y al modelo de datos previamente explicado. La sección que se utilizó como

[1] *JTreeMap* establece un árbol jerárquico donde cada nodo tiene un valor y una clave que lo identifica

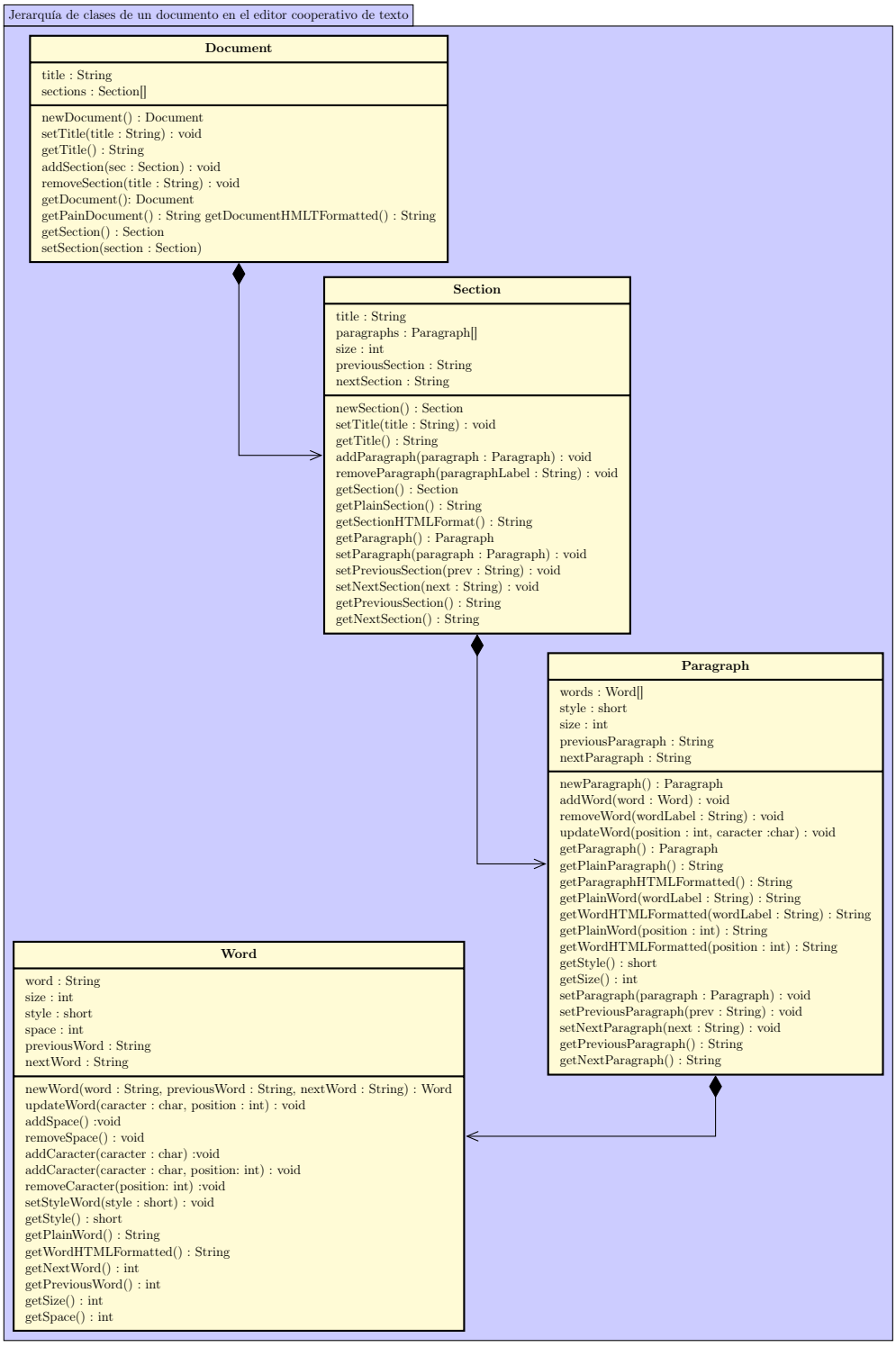


Figura 4.5: Jerarquía de clases de un documento en el editor cooperativo de texto

ejemplo se compone de dos párrafos: el primer párrafo cuenta con diez palabras (formadas por los caracteres 1 al 91); el segundo párrafo cuenta con ventiún palabras (formadas por los caracteres 92 al 224).

Al detectarse un espacio en blanco, se da por finalizada la palabra a la izquierda de dicho espacio y se genera una nueva palabra a su derecha. Si se ingresa más de un espacio en blanco de forma continua, todos estos espacios se consideran parte de la palabra de la izquierda.

Al detectarse un salto de línea, se da por finalizado el párrafo y se genera uno nuevo, el cual incluye una nueva palabra donde continúa escribiendo el usuario.

Paragraph 1	Los	espacios	funcionales	del	sistema	*Groupware*	son:
	Word 1 (1-4)	Word 2 (5-13)	Word 3 (14-25)	Word 4 (26-29)	Word 5 (30-36)	Word 6 (37-46)	Word 7 (47-51)
	Comunicación,	Producción,	Coordinación.				
	Word 8 (52-65)	Word 9 (66-77)	Word 10 (78-91)				
Paragraph 2	La	integración	de	estos	espacios	aumenta	la
	Word 1 (92-95)	Word 2 (96-107)	Word 3 (108-110)	Word 4 (111-116)	Word 5 (117-125)	Word 6 (126-133)	Word 7 (134-136)
	eficacia	de	los	colaboradores	al	trabajar	conjuntamente
	Word 8 (137-145)	Word 9 (146-148)	Word 10 (149-152)	Word 11 (153-166)	Word 12 (167-169)	Word 13 (170-178)	Word 14 (179-192)
	en	la	realización	de	una	tarea	
	Word 15 (193-195)	Word 16 (196-198)	Word 17 (199-210)	Word 18 (211-213)	Word 19 (214-217)	Word 20 (218-223)	Word 21 (224-)

Figura 4.6: Texto de una sección del documento

Los métodos del componente JEditorPane que se utilizaron son: detección de click izquierdo del ratón y detección de la tecla pulsada en el teclado. A continuación, se presentan los algoritmos utilizados para cada método.

La detección de click izquierdo del ratón permite conocer la posición del mismo con respecto al texto, i.e. el caracter en el que se encuentra el cursor dentro del arreglo de JEditorPane. Esta posición se utiliza para identificar la palabra a la que el usuario dió click izquierdo. El algoritmo 2 muestra el procedimiento para dicha identificación.

Algoritmo 2 Detección de la palabra actual y del párrafo actual donde está posicionado el cursor

Entrada: *position*= posición del ratón con respecto al arreglo de caracteres del objeto JEditorPane
numParagraphs = número de párrafos de la sección
iParagraphActual= índice del primer párrafo de la sección
Salida: *iParagraph*= el índice del párrafo que contiene la palabra activa
iWord=el índice de la palabra activa en el párrafo

```
1: size = paragraphs[iParagraphActual].length()
2: mientras position >= size hacer
3:     iParagraphActual = paragraphs[iParagraphActuall].nextParagraph();
4:     size= size+ paragraphs[iParagraphActual].length
5: fin mientras
6: iParagraph = iParagraphActual
7: size = size - paragraphs[iParagraph].length()
8: iWordActual = el índice de la primera palabra de paragraphs[iParagraphActual]
9: mientras position >= size hacer
10:    iWordActual = words[iWordActual].nextWord()
11:    size= size+ words[iWordActual].length()
12: fin mientras
13: iWord = iWordActual
```

La detección de la tecla pulsada en el teclado identifica el tipo de caracter que se ingresó al JEditorPane, se clasifica por: caracter alfanumérico, símbolo, espacio, salto de línea y retorno de carro. A continuación se presenta el algoritmo para cada tipo de caracter que ingresa el usuario, con respecto a la posición del cursor en la palabra actual.

En el algoritmo 3 se presentan las operaciones que se ejecutan para la edición del texto cuando se ingresa un caracter de tipo alfanumérico o símbolo.

Algoritmo 3 Edición de texto al ingresar un caracter de tipo alfanumérico o símbolo

Entrada: *word*= palabra actual sobre la cual se está escribiendo
position= posición donde se encuentra el cursor con respecto a *word*
caracter= letra ingresada desde el teclado sobre el área de trabajo

```
1: si position =1  entonces
2:     // es al inicio del texto de word
3:     word= caracter + word
4: si no, si position > 1 and position<word.length() entonces
5:     //forma parte del texto de word
6:     word1 = word.substring(1,position)
7:     word2 = word.substring(position+1,word.length)
8:     word1 = word1+ caracter
9:     word =word1 +word2
10: si no, si position = word.length() entonces
11:    //es al final del texto de word
12:    word= word+ caracter
13: si no
14:    // se encuentra en los espacios a la derecha de la palabra
15:    space1= position- word.length() // espacios antes de la posición
16:    space2 = word.space()+word.length()-position // espacios después de la posición
17:    word1 = word
18:    word1. addSpace(space1)
19:    word2 = caracter
20:    word2.addSpace(space2)
21: fin si
22: lastCaracterIsSpace = FALSE
```

En el algoritmo 4 se presentan las operaciones que se ejecutan para la edición del texto cuando se ingresa un caracter de tipo espacio.

Algoritmo 4 Edición de texto al ingresar un caracter de tipo espacio

Entrada: *paragraph*= párrafo actual que contiene la palabra que se está editando
word= palabra actual sobre la cual se está escribiendo
position= posición donde se encuentra el cursor con respecto a *word*
caracter= letra ingresada desde el teclado sobre el área de trabajo

```
1: si position =1  entonces
2:     // es al inicio del texto de word
3:     iWordPrev=word.getPreviousWord();
4:     words[iWordPrev].addSpace(1);
5: si no, si position > 1 and position<word.length() entonces
6:     //forma parte del texto de word
7:     word1 = word.substring(1,position)
8:     word2 = word.substring(position+1,word.length)
9:     newWord=paragraph.newWord()
10:    newWord.setWord(word1)
11:    newWord.addSpace(1)
12:    newWord.setPreviousWord(word.getPreviousWord())
13:    newWord.setNextWord(word.getIndex())
14:    word.setWord(word2)
15:    word.setPreviousWord(newWord.getIndex())
16: si no, si position = word.length() entonces
17:    //es al final del texto de word
18:    si lastCaracterIsSpace=FALSE entonces
19:        word.addSpace(1)
20:        word=paragraph.newWord()
21:        newWord.setPreviousWord(word.getIndex()
22:        newWord.setNextWord(word.getNextWord())
23:        word.setNextWord(newWord.getIndex())
24:    si no
25:        iWordPrev=word.getPreviousWord();
26:        words[iWordPrev].addSpace(1);
27:    fin si
28: si no
29:    // se encuentra en los espacios a la derecha de la palabra
30:    word.addSpace(1);
31: fin si
32: lastCaracterIsSpace=TRUE
```

En el algoritmo 5 se presenta las operaciones que se ejecutan para la edición del texto cuando se ingresa un caracter de tipo salto de línea.

Algoritmo 5 Edición de texto al ingresar un caracter de tipo salto de línea

Entrada: *section*=sección actual que se está editando
paragraph= párrafo actual que contiene la palabra que se está editando
word= palabra actual sobre la cual se está escribiendo
position= posición donde se encuentra el cursor con respecto a *word*
caracter= letra ingresada desde el teclado sobre el área de trabajo

```
1: si position =1  entonces
2:     // es al inicio del texto de word
3:     wordResp=word
4:     mientras  wordResp.getNextWord ≠ null hacer
5:         wordsResp.add(word.getNextWord())
6:         paragraph.delete(word.getNextWord()
7:         wordResp=word.getNextWord()
8:     fin mientras
9:     paragraph=section.newPragraph()
10:    word1 =paragraph.addWord()
11:    word1 =word
12:    para todo wordResp ∈ wordsResp hacer
13:        word1 =paragraph.addWord()
14:        word1 =wordResp
15:    fin para
16: si no, si position > 1 and position<word.length() entonces
17:    //forma parte del texto de word
18:    word1 = word.substring(1,position)
19:    word2 = word.substring(position+1,word.length)
20:    word=word1
21:    Se repite este algoritmo de la línea 3 a la 9
22:    word=paragraph.addWord()
23:    word2 =word2
24:    Se repite este algoritmo de la línea 12 a la 15
25: si no, si position = word.length() entonces
26:    //es al final del texto de word
27:    Se repite este algoritmo de la línea 3 a la 9
28:    word=paragraph.addWord()
29:    Se repite este algoritmo de la línea 12 a la 15
30: si no
31:    // se encuentra en los espacios a la derecha de la palabra
32:    Se repite este algoritmo de la línea 3 a la 9
33:    word=paragraph.addWord()
34:    Se repite este algoritmo de la línea 12 a la 15
35: fin si
36: lastCaracterIsSpace=FALSE
```

En el algoritmo 6 se presentan las operaciones que se ejecutan para la edición del texto cuando se ingresa un caracter de tipo retorno de carro.

Algoritmo 6 Edición de texto al ingresar un caracter de tipo retorno de carro

Entrada: *paragraph*= párrafo actual que contiene la palabra que se está editando
word= palabra actual sobre la cual se está escribiendo
position= posición donde se encuentra el cursor con respecto a *word*
caracter= letra ingresada desde el teclado sobre el área de trabajo

1: **si** *position* =1 **entonces**
2: // es al inicio del texto de *word*
3: *iWordPrev*=*word*.getPreviousWord()
4: *words*[*iWordPrev*].deleteSpace(1)
5: **si** *words*[*iWordPrev*].space() = 0 **entonces**
6: *words*[*iWordPrev*]= *words*[*iWordPrev*]+*word*
7: *paragraph*.deleteWord(*word*.getIndex())
8: *word* = *words*[*iWordPrev*]
9: **fin si**
10: **si no**, **si** *position* > 1 and *position*<*word*.length() **entonces**
11: //forma parte del texto de *word*
12: *word1* = *word*.substring(1,*position-1*)
13: *word2* = *word*.substring(*position*+1,*word*.length)
14: *word*=*word1*+ *word2*
15: **si no**, **si** *position* = *word*.length() **entonces**
16: //es al final del texto de *word*
17: *word* = *word*.substring(1,*word*.length()-1)
18: **si no**
19: // se encuentra en los espacios a la derecha de la palabra
20: *word*.deleteSpace(1)
21: Se repite este algoritmo de la línea 5 a la 9
22: **fin si**
23: *lastCaracterIsSpace*=FALSE

Actualización de cambios

Como se explicó en el capítulo anterior, la actualización de cambios se refiere a la forma en que el usuario comparte los cambios que ha realizado con los demás colaboradores y, a su vez, la forma en que éste recibe los cambios que han realizado los demás colaboradores. El controlador del cliente se encarga de administrar el tipo de sincronización que eligió el usuario para cada sección del documento y ejecutar los cambios en el modelo de datos local o global con respecto esta sincronización.

Para compartir y recibir los cambios en la sección, se crearon las clases *CaracterShare* y *WordShare* (véase figura 4.7). Estas clases son las que se serializan mediante *Kryo* y son enviadas a través de *KryoNet*, utilizando el algoritmo 1.

El uso de cada clase depende del tipo de sincronización que elija el usuario, recordando que, para el editor cooperativo de texto, *Misho* ofrece una sincronización síncrona o asíncrona y que la sincronización síncrona depende del tipo de granularidad que elija el usuario: caracter, palabra o párrafo. A continuación, se presentan los procedimientos que

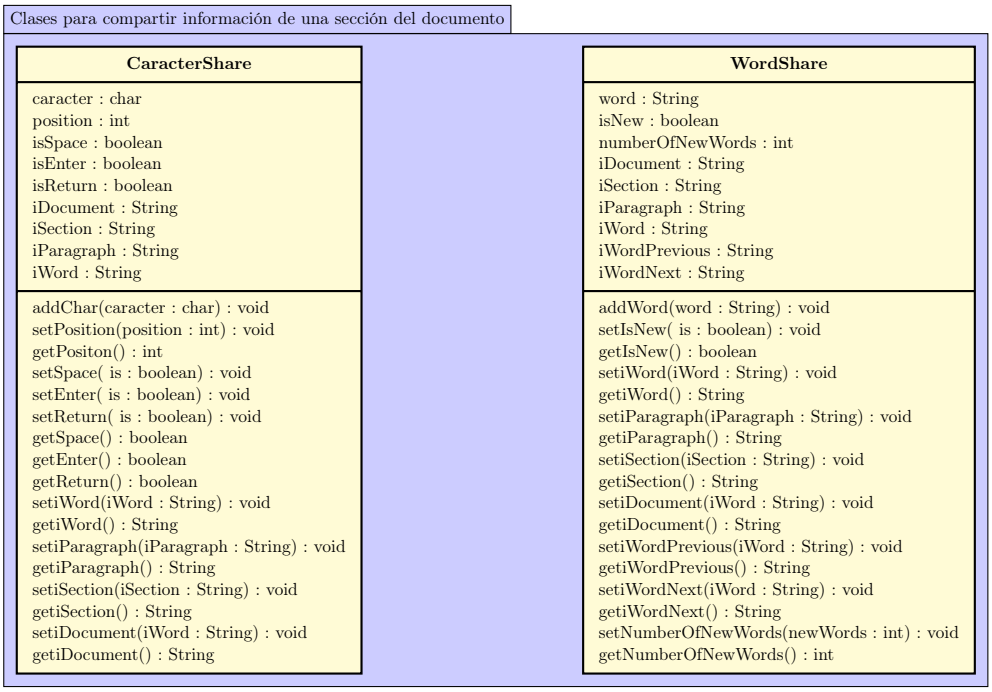


Figura 4.7: Clase para compartir información de una sección del documento

se siguen para cada tipo de sincronización.

Sincronización de tipo síncrona con granularidad de caracter: cuando el controlador del cliente detecta que se ha ingresado un caracter con el teclado, crea un objeto *Obj* de tipo *CaracterShare* y lo envía al servidor utilizando el algoritmo 1. El controlador del cliente también se utiliza cuando el cliente modifica una palabra al ingresar un caracter de tipo retorno de carro o espacio. Cuando el cliente recibe un objeto de tipo *CaracterShare*, lo procesa utilizando el algoritmo 7.

Algoritmo 7 Envío de cambio con granularidad de caracter

Entrada: *Obj*= objeto de tipo *CaracterShare* enviado por el servidor
1: El cliente recibe *Obj* del servidor
2: El controlador obtiene la ubicación del cambio por medio de las variables: *iDocument*, *iSection*, *iParagraph* y *iWord*
3: El controlador obtiene el *caracter* y la *posición* del mismo
4: **si** *caracter* es de tipo alfanumérico o símbolo **entonces**
5: Ejecuta algoritmo 3 para la palabra cuyo índice es *iWord*
6: **si no**, **si** caracter es de tipo espacio **entonces**
7: Ejecuta algoritmo 4 para la palabra cuyo índice es *iWord*
8: **si no**, **si** caracter es de tipo salto de línea **entonces**
9: Ejecuta algoritmo 5 para la palabra cuyo índice es *iWord*
10: **si no**, **si** caracter es de tipo retorno de carro **entonces**
11: Ejecuta algoritmo 6 para la palabra cuyo índice es *iWord*
12: **fin si**
13: Despliega la información en la vista del cliente

Sincronización de tipo síncrona con granularidad de palabra: cuando el controlador del cliente detecta que el caracter ingresado es de tipo espacio o cuando se modifican las palabras en el modelo de datos al ingresar un retorno de carro, crea un objeto *Obj* de tipo *WordShare* y lo envía al servidor utilizando el algoritmo 1. Cuando el cliente recibe un objeto de tipo *WordShare* lo procesa utilizando el algoritmo 8.

Algoritmo 8 Envío de cambio con granularidad de palabra

Entrada: *Obj*= objeto de tipo *WordShare* enviado por el servidor
1: El cliente recibe *Obj* del servidor
2: El controlador obtiene la ubicación del cambio por medio de las variables: el *iDocument*, *iSection* y *iParagraph*
3: El controlador obtiene la *word*
4: **si** *palabra*.isNew() **entonces**
5: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].addWord(*word*)
6: **si no**
7: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].deleteWord(*word*.getIndex())
8: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].addWord(*word*)
9: **fin si**
10: Despliega la información en la vista del cliente

Sincronización de tipo síncrona con granularidad de párrafo: el controlador del cliente almacena todas las palabras que ha ingresado el usuario desde la última actualización y las identifica con una bandera de sincronización desactivada (*isSync*). Cuando se detecta que el caracter ingresado en la sección es de tipo espacio, se verifica el modelo de datos local. Todas las palabras que no tienen la bandera de sincronización activa se envían al servidor, creando un objeto *Obj* de tipo *WordShare* por cada palabra, utilizando el algoritmo 1. El cliente recibe un objeto de tipo *WordShare*, el cual es procesado utilizando el algoritmo 8.

Sincronización de tipo asíncrona: el controlador del cliente espera la petición del usuario para actualizar los datos. En este caso, el controlador del cliente almacena todos los cambios, de la misma forma que la sincronización de tipo síncrona con granularidad de párrafo y en espera de la petición del usuario para enviar todas las palabras que no tienen la bandera de sincronización activa. Se crean los objetos *Obj* de tipo *WordShare* por cada palabra y se utiliza el algoritmo 1 para enviar cada palabra al servidor. El cliente recibe un objeto de tipo *WordShare*, el cual es procesado utilizando el algoritmo 8.

Administración de recursos compartidos

La administración de los recursos compartidos hace referencia al control de concurrencia, utilizando la técnica de bloqueo por granularidad. Para el caso del editor cooperativo de texto, la granularidad de bloqueo es de tres palabras, a las cuales se aplica un candado de edición. Cuando se detecta la palabra que el usuario va a editar, (mediante el algoritmo 2) se envía la petición al servidor para bloquear la palabra actual utilizando el algoritmo 1 para enviar un objeto *Obj* de tipo *DBOperationRequest* (véase figura 4.2) el cual está formado por la clase "Editor", la operación "LockWord" y los parámetros: *iDocument*,

iSección, *iParagraph*, *iWord*, separando cada variable con el caracter "|". Se envía el objeto *Obj* a través de *KryoNet* y se espera la respuesta del servidor utilizando la clase *DBOperationResponse*. Cuando el controlador del cliente recibe la respuesta del servidor, se ejecuta el algoritmo 9.

Algoritmo 9 Candados en una palabra

Entrada: *Obj*= objeto de tipo *DBOperationResponse* enviado por el servidor
1: El controlador recibe *Obj* del servidor
2: El controlador obtiene el contenido de las variables *classes*, *operation*, *response*
3: **si** *classes* = "Editor" **entonces**
4: **si** *operation* = "LockWord" **entonces**
5: El controlador obtiene de *response* los valores de los parámetros *iDocument*, *iSection*, *iParagraph*,*iWord*, *statusWord* y *user*
6: **si** *statusWord* ="lock" **entonces**
7: // Notifica al usuario que la palabra está ocupada por otro colaborar
8: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWord*].updateColorFont(gray)
9: *iWordPrevious*=*documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWord*].getPreviousWord()
10: *iWordNext*=*documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWord*].getNextWord()
11: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWordPrevious*].updateColorFont(gray)
12: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWordNext*].updateColorFont(gray)
13: despliega notificación con el nombre del usuario que tiene el permiso de modificación (*user*)
14: wordIsLocked=TRUE
15: **si no**
16: wordIsLocked=FALSE
17: **fin si**
18: **si no**, **si** *operation* = "UnlockWord" **entonces**
19: // Notifica al usuario que la palabra está disponible para edición
20: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWord*].updateColorFont(black)
21: *iWordPrevious*=*documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWord*].getPreviousWord()
22: *iWordNext*=*documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWord*].getNextWord()
23: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWordPrevious*].updateColorFont(black)
24: *documents*[*iDocument*].*sections*[*iSection*].*paragraphs*[*iParagraph*].*words*[*iWordNext*].updateColorFont(black)
25: **fin si**
26: **fin si**
27: Despliega la información en la vista del cliente

La duración del candado depende del tipo de sincronización que maneje el usuario en la sección, i.e. el candado permanece activo hasta que se actualice la información al servidor. Cuando esto ocurre, se envía una petición de desbloqueo utilizando la clase *DBOperationRequest*.

Modificación del estilo de la fuente del texto

Los objetos de tipo *Word* almacenan el texto plano de la palabra y un conjunto de banderas que permiten conocer el estilo que tiene dicha palabra, lo cual hace sencillo exportar el estilo de la palabra a diferentes formatos (HTML y PDF).

Para modificar el estilo de la fuente del texto de la sección activa, el usuario debe seleccionar una acción de la barra de herramientas del editor cooperativo de texto. Cuando se detecta una acción de esta barra, se verifica si existe un fragmento de texto seleccionado al que se desea aplicar dicha modificación, en caso contrario, se considera como una modificación de estilo para las palabras que posteriormente ingrese el usuario. La modificación del estilo de las palabras se aplica utilizando el algoritmo 10 el cual modifica las banderas *style*, *font*, *fontSize*, *colorFont* y *colorBackground*.

Algoritmo 10 Modificación de estilo de la fuente del texto

Entrada: *iWordInit*= índice de la palabra inicial del texto seleccionado
iWordFinish= índice de la palabra final del texto seleccionado
style= estilo de la palabra (negrita, cursiva, subrayado, subíndice, superíndice)
font = tipo de fuente de la palabra
fontSize= tamaño de la fuente de la palabra
colorFont = color de la fuente de palabra

1: **si** *iWordInit* = *iWordFinish* **entonces**
2: Se mantiene el estilo indicado como global para las palabras que posteriormente ingrese el usuario
3: **fin si**
4: **mientras** *wordInit* ≠ *wordFinish* **hacer**
5: Se modifica el estilo de la fuente de letra de *iWordInit*
6: **si** tipo de modificación =*style* **entonces**
7: *words*.[*iWordInit*].addStyle(*style*)
8: **si no**, **si** tipo de modificación =*font* **entonces**
9: *words*.[*iWordInit*].setFont(*font*)
10: **si no**, **si** tipo de modificación =*fontSize* **entonces**
11: *words*.[*iWordInit*].setSizeFont(*fontSize*)
12: **si no**, **si** tipo de modificación =*colorFont* **entonces**
13: *words*.[*iWordInit*].setColorFont(*colorFont*)
14: **fin si**
15: *iWordInit* =*words*.[*iWordInit*].getNextWord()
16: **fin mientras**
17: Se repite este algoritmo de la línea 5 a la 14
18: Despliega la información en la vista del cliente

De igual forma, los objetos de tipo *Paragraph* utilizan una bandera que permite identificar la alineación que tiene dicho párrafo, con la finalidad de permitir la alineación independiente de los párrafos. Cuando se detecta la acción de cambio en la alineación del párrafo desde la barra de herramientas, se utiliza el algoritmo 11 para cambiar dicha bandera dependiendo el tipo de alineación seleccionada (*center*, *left*, *right* o *justified*).

Algoritmo 11 Modificación de la alineación del párrafo

Entrada: *iParagraph*= índice del párrafo seleccionado
center= alineación centralizada
right = alineación a la derecha
left= alineación a la izquierda
justified = alineación justificada

```
1: si tipo de modificación =center entonces
2:     paragraphs.[iParagraph].setAling(center)
3: si no, si tipo de modificación =right entonces
4:     paragraphs.[iParagraph].setAling(right)
5: si no, si tipo de modificación =left entonces
6:     paragraphs.[iParagraph].setAling(left)
7: si no, si tipo de modificación =justified entonces
8:     paragraphs.[iParagraph].setAling(justified)
9: fin si
10: Despliega la información en la vista del cliente
```

Soporte de *deixis*

Previamente a la petición de *deixis*, el usuario debe seleccionar el fragmento de texto que desea compartir. Cuando el usuario indica que quiere compartir el texto seleccionado, el controlador del cliente solicita, con un cuadro de diálogo, el nombre de la etiqueta del hipervínculo que hace referencia al texto seleccionado. Una vez que el usuario ingrese la etiqueta, el controlador genera un hipervínculo en la sala de mensajería activa. El hipervínculo contiene la ubicación del texto seleccionado de la forma "Documento|Sección|Párrafo|PalabraInicial|PalabraFinal". Para detectar cuales son las palabras inicial y final de la sección, se utiliza el algoritmo 2.

Cuando el usuario da click en el hipervínculo de la sala de mensajería, se toma la ubicación del texto seleccionado del hipervínculo y se modifica el estilo de las palabras del texto de la sección, al cambiar el color del fondo del texto (*colorBackground*) a cyan utilizando el algoritmo 12.

Algoritmo 12 Soporte de *deixis*

Entrada: *iWordInit*= índice de la palabra inicial del texto seleccionado
iWordFinish= índice de la palabra final del texto seleccionado
colorBackground= color del fondo del texto de la palabra

```
1: mientras wordInit ≠ wordFinish hacer
2:     words[iWordInit].setBackground(colorBackground)
3:     iWordInit=words[iWordInit].getNextWord()
4: fin mientras
5:
6: words[iWordInit].setBackground(colorBackground)
7: Despliega la información en la vista del cliente
```

El texto permanece sombreado de color cyan hasta que el usuario realice una acción diferente en la sección del documento. En ese momento se ejecuta el algoritmo 12 para cambiar el color de fondo del texto a negro.

4.2.2. Pizarrón cooperativo

El pizarrón cooperativo está compuesto por: 1) una vista que permite la creación, administración y edición de imágenes; 2) un modelo de datos que agrupa jerárquicamente la información perteneciente las imágenes que cada usuario está utilizando en cada sesión; y finalmente, 3) un controlador, que recibe y procesa las acciones que realiza el usuario sobre la vista para modificar o consultar el modelo de datos de forma local o global.

La vista del editor cooperativo de texto se muestra en la figura 4.8. Está compuesta por: 1) una lista de imágenes (conjunto de pestañas) en la parte inferior de la ventana y 2) un panel de edición de la imagen, el cual contiene la barra de herramientas y el lienzo.

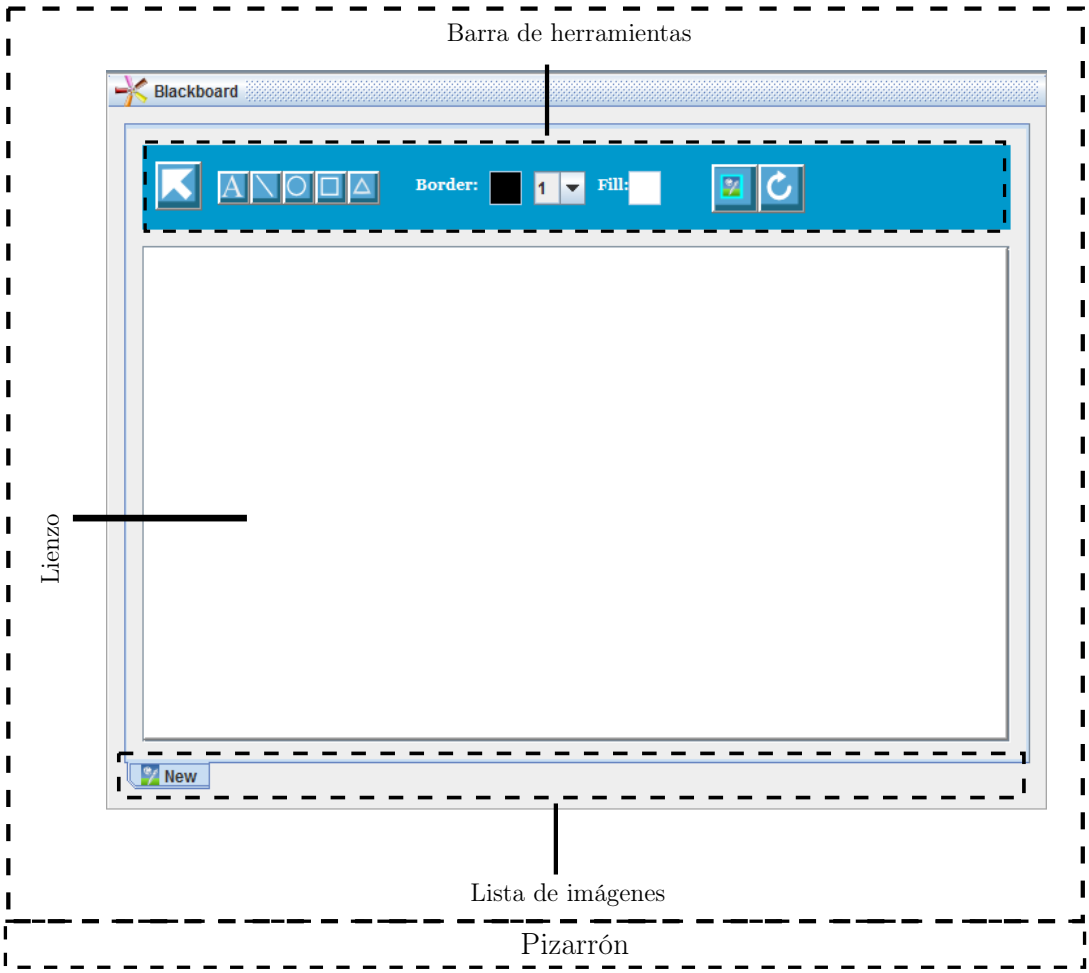


Figura 4.8: Interfaz gráfica de la aplicación Pizarrón

Las acciones que puede realizar cada usuario sobre la vista del editor cooperativo de texto son:

1 *Crear una nueva imagen*
Esta acción crea una nueva imagen perteneciente a un grupo de trabajo específico.

2 *Abrir una imagen existente*
Esta acción carga las figuras pertenecientes a la imagen seleccionada en el lienzo. Utiliza la barra de herramientas para modificar las propiedades de la figura activa. Las opciones que ofrece la barra de herramientas son:

a) Cambiar el tamaño del ancho del borde de la figura selecciona.

b) Cambiar el color del borde de la figura seleccionada.

c) Cambiar el color del relleno de la figura seleccionada.

d) Compartir una figura de la imagen.

e) Aprobar la imagen activa[1].

f) Sincronizar información de forma síncrona o asíncrona con el servidor[2].

3 *Guardar la imagen activa*
Esta acción guarda todos los cambios realizados en la imagen activa.

4 *Cerrar la imagen activa*
Esta acción cierra la imagen activa. En caso de existir cambios, se notifica al usuario si se desean guardar.

5 *Exportar la imagen activa*
Esta acción permite exportar la imagen a un archivo con formato JPG y PNG.

El modelo de datos del pizarrón cooperativo se construyó de tal forma que cuenta con una lista de imágenes, donde cada imagen está dividida en una lista de figuras. Para organizar y almacenar la información de cada imagen se utilizó un objeto de tipo *JTreeMap* de Java. Este objeto permite agilizar la búsqueda de las figuras que contiene el pizarrón, al utilizar solo la clave que identifica a la figura que se busca. En la figura 4.9, se presenta la jerarquía de clases de un figura en el pizarrón cooperativo.
El controlador del pizarrón cooperativo se divide en tres áreas con las que trabaja conjuntamente, las cuales son: 1) recibe las peticiones del usuario a través de la vista, 2) manipula el contenido del modelo de datos local y 3) comunica las acciones al servidor. Los métodos presentados en la figura 4.9 representan la parte del controlador que permite manipular la información de una imagen.

[1] en caso de contar con el rol
[2] en caso de tener seleccionada una sincronización asíncrona en la sección

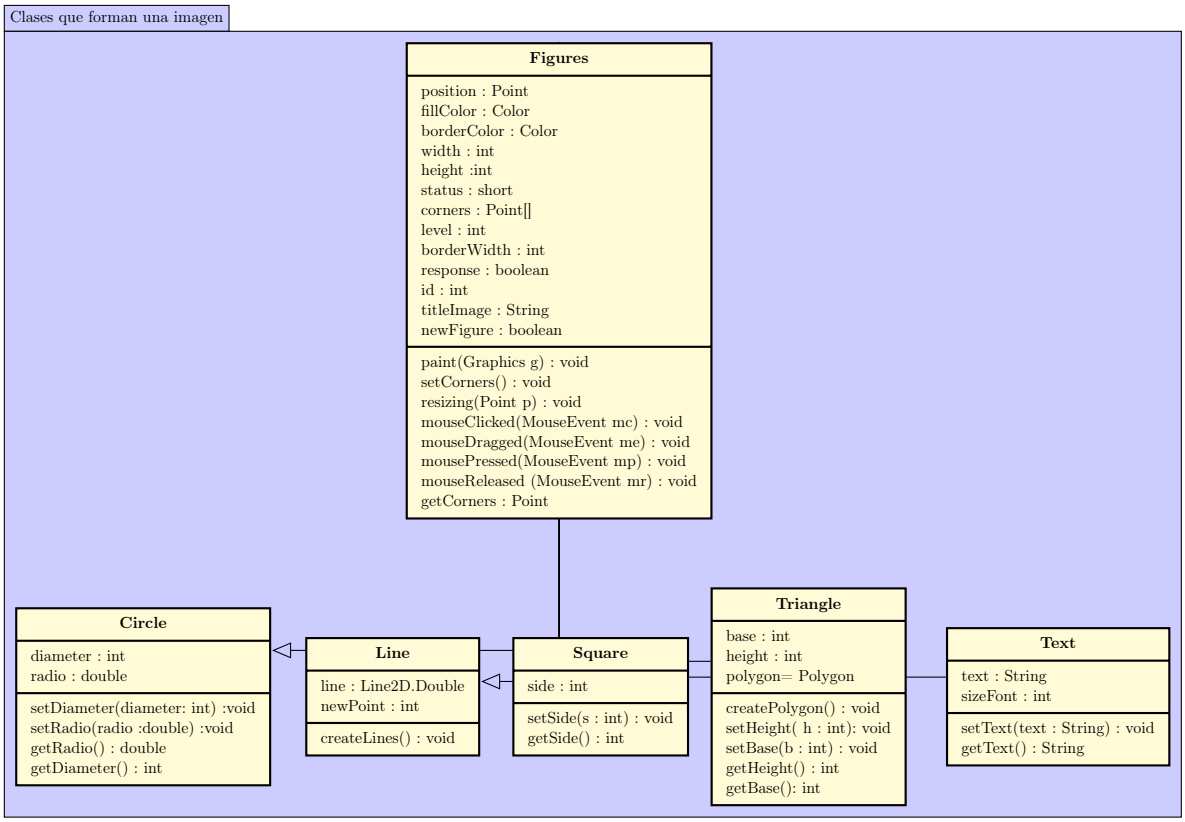


Figura 4.9: Clases que forman una imagen

Como se explicó en el capítulo anterior, en la edición cooperativa de una imagen se requiere: 1) edición de imagen; 2) actualización de cambios, 3) administración de recursos compartidos, 4) modificación de las propiedades de la figura y 5) soporte de *deixis*.

Edición de una imagen

El lienzo está compuesto por un componente *JPanel* de Java, al que denominamos *blackboard*, por medio del cual se selecciona la posición de la figura y al mismo tiempo se despliegan las figuras que componen de dicha imagen. Los métodos del componente *JPanel* que se utilizan son: detección del click izquierdo del ratón, detección del arrastre de ratón y detección de cuando se suelta el click izquierdo del ratón.

Para agregar una figura, se debe seleccionar en la barra de herramientas la figura deseada y posteriormente hacer click izquierdo en el lienzo. De esta forma, se obtiene la posición del ratón con respecto al lienzo del pizarrón donde el controlador debe agregar la figura.

Para seleccionar una figura, se debe seleccionar, en la barra de herramientas, la acción

de selección y posteriormente hacer click izquierdo con el ratón sobre la figura. Si existe más de una imagen presente en esa posición, el controlador del cliente toma la última que se colocó en dicha posición. Cuando se selecciona una figura, se pinta un cuadro de edición que permite escalar y trasladar la figura, siguiendo el desplazamiento del ratón con respecto al lienzo.

Sincronización de cambios

Como se explicó en el capítulo anterior, la actualización de cambios se refiere a la forma en que el usuario comparte los cambios que ha realizado con los demás colaboradores y, a su vez, la forma en que el usuario recibe los cambios que han realizado los demás colaboradores. El cliente solo se encarga de administrar el tipo de sincronización que eligió el usuario para cada imagen y de ejecutar los cambios en el modelo de datos local o global con respecto a esta sincronización.
Misho ofrece una sincronización de tipo síncrona o asíncrona para actualizar los cambios en las figuras de la imagen. A continuación, se presentan los procedimientos que se siguen para cada tipo de sincronización.
Sincronización síncrona: cuando el controlador del cliente detecta que se ha agregado o modificado una figura, envía este objeto de clase *Figure* previamente serializado por *Kryo* al servidor, utilizando la biblioteca *KryoNet* y el algoritmo 1. Cuando el cliente recibe un objeto de tipo *Figure* lo procesa utilizando el algoritmo 13.

Algoritmo 13 Envío de figuras

Entrada: *Obj*= objeto de tipo *Figure*
1: El controlador recibe el objeto *Obj* del servidor
2: **si** figure.isNew()=TRUE **entonces**
3: *figures*.addFigure(*Obj*)
4: **si no**
5: *figures*.deleteFigure(*Obj*.getIndex())
6: *figures*.addFigure(*Obj*)
7: **fin si**
8: Despliega la información en el lienzo

Sincronización asíncrona: cuando el usuario solicita la sincronización de la información que ha actualizado,el controlador del cliente almacena todas las figuras que ha ingresado el usuario desde la última actualización y las identifica con la bandera de sincronización desactivada (*isSyncFigure*). Cada una de estas figuras no actualizadas es serializada a un objeto de tipo *Figure* y enviada al servidor usando el algoritmo 1. Finalmente, los clientes reciben las figuras no actualizadas como objetos de tipo *Figure* y posteriormente los agregan a su modelo de datos local, utilizando el algoritmo 13.

Administración de recursos compartidos

La administración de los recursos compartidos hace referencia al control de concurrencia utilizando la técnica de bloqueo por granularidad. Para el caso del pizarrón cooperativo, la granularidad a la que se aplica un candado es de nivel figura. Cuando se selecciona una figura del lienzo, se envía la petición al servidor para bloquearla utilizando un objeto de tipo *DBOperationRequest* (véase figura 4.2) el cual está formado por la clase "Blackboard", la operación "lockFigure" y los parámetros: "*iImage* | *iFigure* | *statusFigure* |*user*". El controlador del cliente envía el objeto al servidor a través de *KryoNet* y se espera la respuesta del servidor. Cuando el controlador del cliente recibe la clase de tipo *DBOperationResponse* se ejecuta el algoritmo 14.

Algoritmo 14 Candado en figura

Entrada: *Obj*= objeto de tipo *DBOperationResponse* enviado por el servidor
1: El controlador recibe el objeto *Obj* del servidor
2: El controlador obtiene el contenido de las variables *classes*, *operation*, *response*
3: **si** *classes* = "Blackboard" **entonces**
4: **si** *operation* = "LockFigure" **entonces**
5: El controlador obtiene de response los valores de los parámetros: *iImage*,*iFigure*, *statusFigure* y *user*
6: **si** *statusFigure* = "lock" **entonces**
7: // Notifica al usuario que la figura está ocupada por otro colaborar
8: *images*[*iImage*].*figures*[*iFigure*].setColorBoder(gray)
9: *images*[*iImage*].*figures*[*iFigure*].setColorFill(gray)
10: despliega notificación con el nombre del usuario que tiene el permiso de modificación (*user*)
11: **si no**, **si** *statusFigure* = "unlock" **entonces**
12: // Notifica al usuario que la figura está disponible para ser editada
13: *images*[*iImage*].*figures*[*iFigure*].setColorBoder(*original*)
14: *images*[*iImage*].*figures*[*iFigure*].setColorFill(*original*)
15: *figureIsLocked*=FALSE
16: **fin si**
17: **fin si**
18: **fin si**
19: Despliega la información en el lienzo

La duración del candado depende del tipo de sincronización que el usuario utilice en la imagen, permaneciendo activo hasta que se actualice la información en el servidor. Cuando se actualiza la información en el servidor, se envía una petición de desbloqueo de dicha figura, utilizando la clase *DBOperationRequest*.

Modificación de las propiedades de una figura

Cuando se selecciona un cambio en las propiedades de la figura, se verifica si existe una figura seleccionada y se aplica el cambio de la propiedad por medio de las variables de la figura *borderWidth*, *borderColor* y *fillColor*, utilizando el algoritmo 15. En caso de no haber seleccionado una figura, el cambio realizado en las propiedades se conserva para las siguientes figuras que ingresen al lienzo.

Algoritmo 15 Modificación de las propiedades de la figura

Entrada: *iFigure*= índice de la figura seleccionada
borderWidth= tamaño del ancho del borde de la figura
bordeColor = color del borde de la figura
fillColor= color del relleno de la figura

```
1: si iFigure ≠ null entonces
2:     Se modifica la propiedad de la figura cuyo indice es iFigure
3:     si tipo de modificación = borderWidth entonces
4:         figures[iFigure].setBorderWidth(borderWidth)
5:     si no, si tipo de modificación = borderColor entonces
6:         figures[iFigure].setBorderColorborderColor)
7:     si no, si tipo de modificación = fillColor entonces
8:         figures[iFigure].setFillColorfillColor)
9:     fin si
10: si no
11:     Se mantiene la propiedad indicada como global para las figuras que posteriormente ingrese el usuario
12: fin si
13: Despliega la información en el lienzo
```

Soporte de *deixis*

Previamente a la petición de *deixis*, el usuario debe seleccionar la figura que desea compartir. Cuando se detecta la acción de compartir, el controlador del cliente solicita mediante un cuadro de diálogo, el nombre de la etiqueta del hipervínculo que hace referencia a la figura seleccionada. Después de que el usuario ingrese la etiqueta, el controlador genera un hipervínculo en la sala de mensajería activa. El hipervínculo contiene la ubicación de la imagen seleccionada de la forma "*iImagen*| *iFigura*".

Cuando el usuario realiza un click izquierdo en el hipervínculo de la sala de mensajería, se toma la ubicación de la imagen seleccionada del hipervínculo y se modifica las propiedades de la misma. Se cambia el color del borde a cyan mediante un temporizador que alterna el cambio del color del borde entre cyan y el color original de la figura, dando la percepción al usuario de que parpadea la figura. Este proceso se describe en el algoritmo 16.

Algoritmo 16 Soporte de *deixis*

Entrada: *iFigura*= índice de la figura seleccionada
isDeixis= bandera que indica que existe una deixis sobre la figura *colorBordeOriginal*= color original del borde de la figura seleccionada

```
1: timerBlink.setPeriod(.5)
2: timerBlink.start()
3: mientras isDeixis =TRUE hacer
4:     si blink=FALSE entonces
5:         figures.figures[iFigures].setColorBoder(cyan)
6:         blink=TRUE
7:     si no
8:         figures.figures[iFigures].setColorBoder(colorBordeOriginal)
9:         blink=FALSE
10:    fin si
11:    Despliega la información en el lienzo
12: fin mientras
13: Se repite este algoritmo de la línea 8 y 9
14: Despliega la información en el lienzo
```

4.2.3. Sala de mensajería instantánea

La sala de mensajería instantánea está compuesta por: 1) una vista que permite la comunicación entre los usuarios a través del envío de mensajes de texto y la recepción de los mismos; 2) un modelo de datos que almacena todos los mensajes generados en cada sala de conversación, 3) un controlador que se encarga de enviar y recibir los mensajes de texto.

La vista de la sala de mensajería instantánea se muestra en la figura 4.10. Esta vista está compuesta por: 1) una lista de usuarios (conjunto de pestañas) en la parte izquierda de la ventana, en donde se puede tener una conversación por cada par de usuarios o de forma general con todos los usuarios de la sesión, 2) un área de mensaje de texto para ingresar el texto del mensaje, 3) un área de visualización de mensajes de texto y 4) un botón para enviar el mensaje de texto.

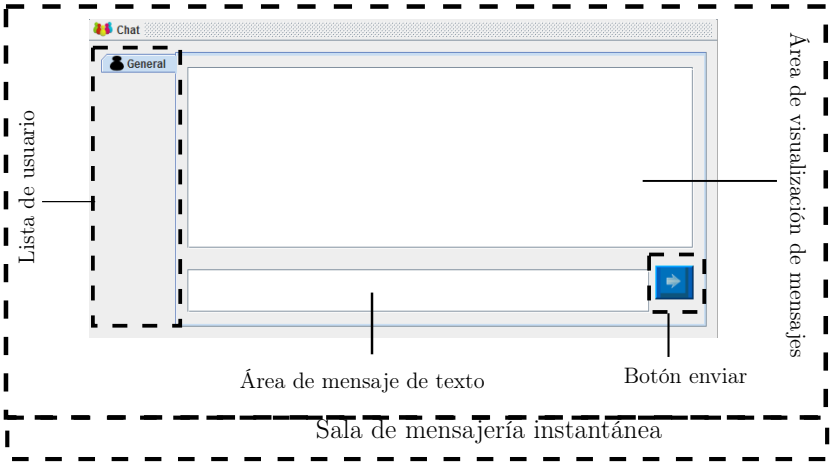


Figura 4.10: Interfaz gráfica de la aplicación Sala de mensajería instantánea

La lista de usuarios conectados se actualiza conforme los usuarios inician sesión al servidor. El servidor envía un objeto de tipo *Users*, el cual contiene la lista de usuarios conectados. El cliente al recibir un objeto de tipo *Users* actualiza las pestañas de la sala de mensajería.

El área de mensaje de texto y el área de visualización de mensajes son objetos de tipo *JEditorPane*. El primero permite la edición del objeto, mientras que el segundo sólo visualiza la información de los mensajes recibidos. El área de visualización de mensajes también permite realizar un click izquierdo con el ratón al hipervínculo que contiene la referencia *deixis* al espacio de producción.

Cada conversación tiene dos procesos: **enviar mensajes** y **visualizar mensajes**. Para ambos procesos se creó una clase *ChatMessengerRequest* (véase figura 4.11), la cual se se-

rializa mediante *Kryo* y permite enviar los mensajes entre el cliente y el servidor a través de *KryoNet* (utilizando el algoritmo 1).

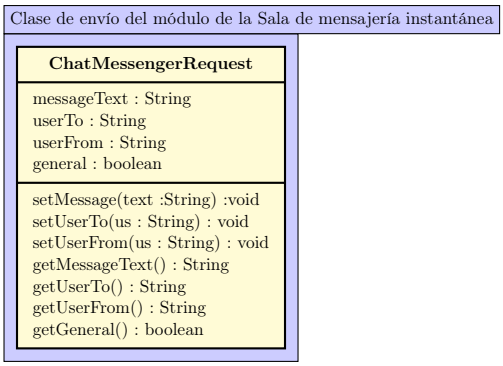


Figura 4.11: Clase de envío del módulo de la Sala de Mensajería Instantánea

Cuando el usuario solicita enviar un mensaje en una de las salas de conversación, se crea un objeto de tipo *ChatMessengerRequest* con el contenido del mensaje y se envía a través de *KryoNet*. El controlador del cliente espera la recepción de dicha clase como respuesta del servidor y ejecuta el algoritmo 17.

El controlador del cliente al detectar que el objeto enviado es de tipo *ChatMessageRequest*, toma el mensaje que fue enviado y lo presenta el área de visualización de mensajes de la pestaña que le corresponde, dependiendo del remitente del mensaje.

Algoritmo 17 Recepción de mensaje de la sala de mensajería instantánea

Entrada: *Obj*= objeto de tipo *ChatMessengerRequest* enviado por el servidor
user= usuario receptor del mensaje

1: El controlador recibe el objeto *Obj* del servidor
2: El controlador obtiene los valores de las variables *userTo*, *userFrom*, *isGeneral* y *message*
3: **si** isGeneral=TRUE **entonces**
4: *conversation*="General"
5: **si no**
6: **si** *userFrom* $\neq$ *user* **entonces**
7: *conversation*= *userFrom*
8: **si no**, **si** *userTo* $\neq$ *user* **entonces**
9: *conversation*= *userTo*
10: **fin si**
11: **fin si**
12: *conversation*.addMessage(*message*)
13: Despliega información de la conversación

4.2.4. Operatividad de *Misho*

Este apartado presenta otras funcionalidades adicionales que ofrece *Misho*, las cuales son: 1) registro de usuarios, 2) inicio de sesión, 3) creación de grupos de trabajo, y 4) invitación colaboradores. Para que los colaboradores interactúen con estas funcionalidades, estas se presentan en un cuadro de diálogo.

Registro usuario

El registro del usuario se realiza mediante un cuadro de diálogo que solicita al usuario que ingrese sus datos, tales como: clave de usuario (*user*), contraseña (*password*), nombre (*name*) y correo electrónico (*email*). Cuando el usuario envía la petición de registrar usuario, se crea objeto de tipo *DBOperationRequest* y se utiliza el algoritmo 1. El controlador del cliente permanece en espera de la respuesta de un objeto de tipo *BDOperationResponse* donde el servidor informa si el usuario fue registrado o existió algún problema en su registro, mismo que se notifica al usuario mediante un cuadro de diálogo.

Inicio de Sesión

El inicio de sesión presenta un cuadro de diálogo que solicita al usuario que ingrese los datos clave de usuario (*user*) y contraseña (*password*) para enviar la petición de autenticación al servidor, utilizando un objeto de tipo *DBOperationRequest* y el algoritmo 1. El controlador del cliente permanece en espera de la respuesta de un objeto de tipo *BDOperationResponse*, donde el servidor informa si el usuario a sido autentificado y se puede dar acceso a la visualización de *Misho*. En caso que no se haya autentificado el usuario, se le notifica sobre el dato erróneo. Después de tres intentos fallidos, se presenta un diálogo con un objeto *captcha*[1], con el fin de evitar ataques de software malicioso.

Creación de grupos de trabajo

Todos los usuarios de *Misho* por defecto pertenecen al grupo de trabajo denominado G_{public}. Sin embargo, para manejar diferentes documentos o imágenes y controlar el acceso a estos, se optó por incluir la creación de grupos de trabajo en *Misho*. Un ejemplo de una sub-agrupación de usuarios que se puede crear en *Misho* se presenta en la figura 4.12. En esta figura se considera que *Misho* cuenta con cuatro usuarios registrados (usuarios

[1] *Captcha* es una imagen que contienen una secuencia de caracteres alfanuméricos aleatorios, mismos que se solicita que el usuario identifique e ingrese en un cuadro de texto que permite validar que la petición de inicio de sesión está siendo realizada por un usuario y no un robot

A ,B, C y D). Los grupos de trabajo a los cuales pertenecen los usuarios son los siguientes:

$$\begin{aligned} usuarios \quad & \{A, B, C, D\} \in G_{public} \\ usuarios \quad & \{A, B\} \in G_1 \\ usuarios \quad & \{B, C\} \in G_2 \\ usuarios \quad & \{A, C\} \in G_3 \end{aligned}$$

Donde todos los usuarios pueden acceder a la información asociada a G_{public} y los usuarios A, B, C pueden acceder a la información de los grupos a los que están inscritos, de forma tal que el usuario D no accede a la información asociada a G_1, G_2 o G_3, ya que no está inscrito a ellos.

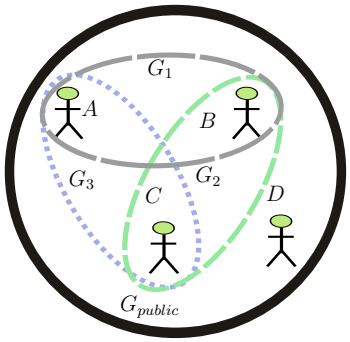


Figura 4.12: Grupos de trabajo

Cuando el usuario solicita crear un grupo de trabajo, se despliega un cuadro de diálogo que presenta: 1) una lista de grupos de trabajo, 2) una lista de los usuarios de *Misho* que no están inscritos al grupo seleccionado y 3) una lista de los usuarios que están inscritos al grupo seleccionado. Al seleccionar un grupo de trabajo, automáticamente se carga la información perteneciente a cada lista de usuarios.

El cuadro de diálogo de grupos de trabajo permite: 1) agregar un grupo de trabajo, 2) invitar usuarios y 3) eliminar una invitación de un usuario. A continuación, se describe el proceso que sigue la acción agregar usuario y en el apartado siguiente se describen los procesos que siguen las demás acciones.

Agregar un grupo de trabajo

Para agregar un grupo de trabajo, el usuario ingresa el nombre del grupo (*groupName*) en un campo de texto y solicita agregar dicho grupo. El controlador de cliente crea un objeto

de tipo *DBOperationRequest* con dicha petición y lo envía al servidor, utilizando el algoritmo 1. El controlador del cliente permanece en espera de la respuesta de un objeto de tipo *BDOperationResponse* donde el servidor informa si el grupo de trabajo fue registrado o existió algún problema durante su registro, mismo que se notifica al usuario mediante un mensaje de diálogo. En caso de que se haya registrado el grupo, éste se agrega a la lista de grupos de trabajo.

Invitación de usuarios al grupo de trabajo

La acción de invitación de usuario permite invitar o retirar la invitación a un usuario. Esta acción es utilizada en la administración de acceso a grupos de trabajo, documentos, secciones de un documento e imágenes. Para invitar a un usuario, se selecciona su nombre en la lista de usuarios de *Misho* y se da click izquierdo con el ratón en el botón *invite*. Para retirar una invitación a un usuario, se selecciona su nombre de la lista de usuarios del grupo de trabajo y se da click izquierdo con el ratón en el botón *uninvite*. En ambas peticiones, el controlador del cliente crea un objeto de tipo *DBOperationRequest* con la petición correspondiente y lo envía al servidor utilizando el algoritmo 1. El nombre del usuario se elimina de la lista seleccionada y se agrega a la lista correspondiente dependiendo si se invita o anula la invitación a un usuario.

4.3. Servidor de *Misho*

Como se explicó en el capítulo anterior, el servidor de *Misho* centraliza la información global generada en el entorno, i.e. el servidor almacena, administra y controla la información de: los usuarios, los grupos de trabajo, los documentos, las imágenes, los roles y permisos que tiene cada usuario sobre cada documento o imagen. Para ello, el servidor requiere un controlador y un modelo de datos globales.

El modelo de datos global está compuesto por: 1) la estructura jerárquica de objetos Java (descrita en la subsección 4.2) que almacena la información de los documentos e imágenes y 2) una base de datos[1] que almacena los datos asociados a los permisos y accesos que tienen los usuarios sobre los datos de la estructura jerárquica de objetos Java.

La información de la base de datos se utiliza para controlar y administrar las sesiones activas de los usuarios. El diagrama entidad-relación de la base de datos del servidor *Misho* se presenta en la figura 4.13.

[1] La base de datos se crea y configura en la primera ejecución del servidor.

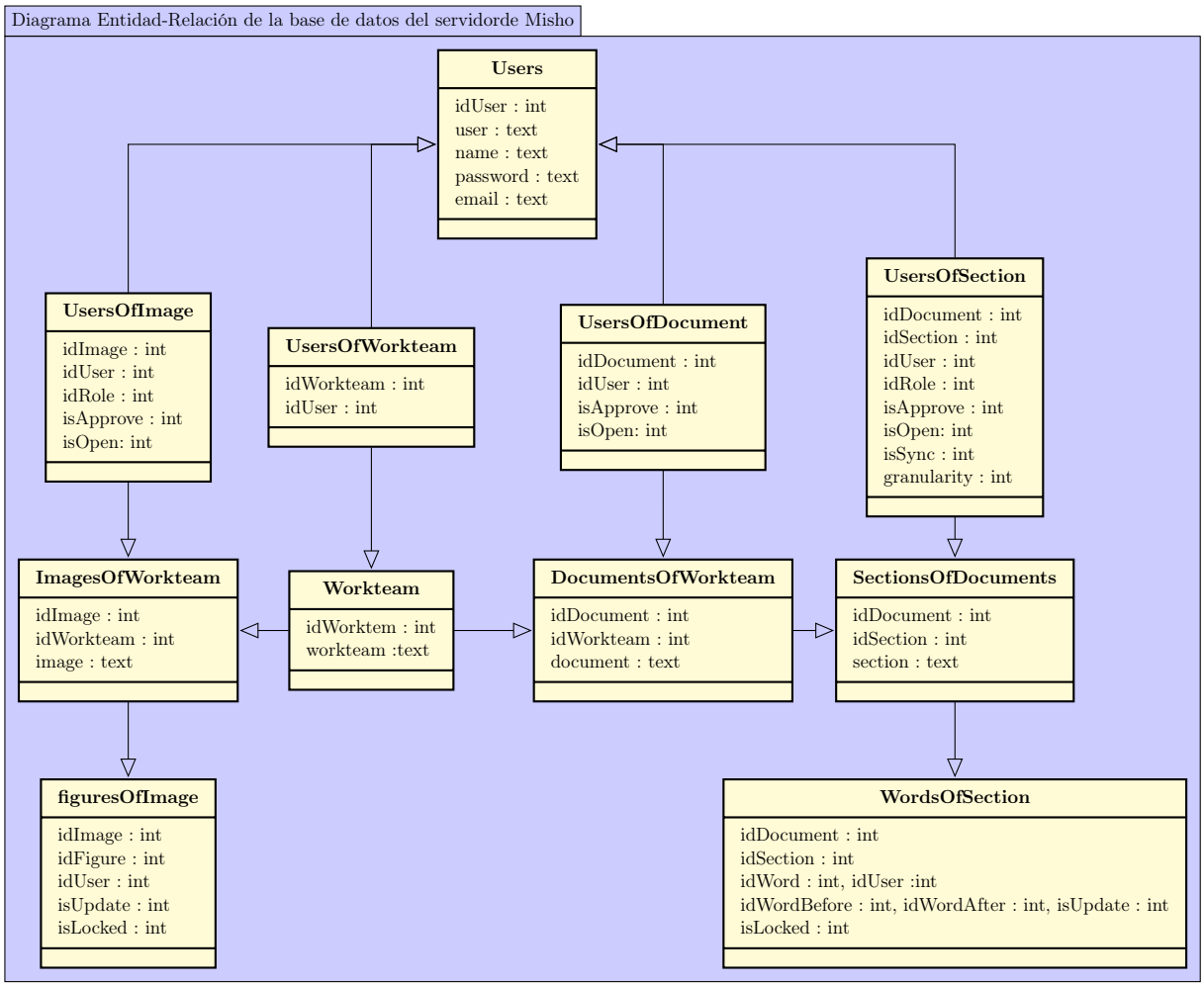


Figura 4.13: diagrama entidad-relación de la base de datos del servidor de *Misho*

A continuación se explicaran las peticiones que modifican los datos de la base de datos.

1 *Consultas a base de datos*
Responde a todas las peticiones de las clases *DBOperationRequest* y *DBOperation-Response*

2 *Registro de usuario*
Al registrarse un usuario se agrega un registro a la tabla *User* y se incluye al grupo de trabajo *public* con el fin que se integre a las actividades de dicho grupo de trabajo, agregando un registro a la tabla *UsersOfWorkteam*.

3 *Agregar Grupo de trabajo*
Al agregar un grupo de trabajo, se agrega un registro a la tabla *Workteam*

4 *Invitar usuarios al grupo de trabajo*
Se agrega o elimina un registro de la tabla *UsersOfWorkteam*

5 *Editor cooperativo de texto*

5.1 *Abrir documento*
Al abrir un documento se modifica el registro de las tablas *UsersOfDocumento* y *UsersOfSection*, indicando que está siendo utilizado por el usuario

5.2 *Cambiar de tipo de sincronización*
Al recibir una petición de cambio de sincronización, se modifica el registro de la tabla *UsersOfSection* indicando que está siendo utilizado por el usuario de forma síncrona o asíncrona

5.3 *Guardar documento*
Al guardar un documento nuevo se agrega un registro en la tabla *DocumentsOfWorkteam* y un registro a la tabla *SectionsOfDocument*. asimismo por cada palabra que tiene la sección del documento se agrega un registro en la tabla *WordsOfSection*. En caso de guardar un documento existente sólo se modifican los registros de la tabla *WordOfSection* y se asigna el usuario que lo creó, agregando un registro en las tablas *UsersOfDocument*, *UsersOfSection*.

5.4 *Crear sección de documento*
Al crear una sección de un documento, se agrega un registro en la tabla *SectionsOfDocument* y se asigna el usuario que lo creó, agregando un registro en la tabla *UsersOfSection*.

5.5 *Invitar usuarios a la sección del document*
Se agrega o elimina un registro de la tabla *UsersOfSection*

5.6 *Crear o modificar una palabra de la sección*
Al recibir una petición que modifique una palabra del texto de la sección del documento, se agrega o actualiza el registro en la tabla *WordsOfSection*

6 *Pizarrón cooperativo*

6.1 *Abrir imagen*
Al abrir una imagen se modifica el registro de la tabla *UsersOfImage* y indicando que está siendo utilizada por el usuario

6.2 *Cambiar de tipo de sincronización*
Al recibir una petición de cambio de sincronización se modifica el registro de la tabla *UsersOfImage*, indicando que está siendo utilizada por el usuario de forma síncrona o asíncrona

6.3 *Guardar imagen*
Al guardar una imagen nueva se agrega un registro en la tabla *ImageOfWorkteam*. asimismo por cada figura que tiene la imagen se agrega un registro en la tabla *figuresOfImage*. En caso de guardar una imagen existente solo se modifican los registros de la tabla *figuresOfImage* y se asigna el usuario que lo creó agregando un registro en las tablas *UsersOfImage*

6.4 *Invitar usuarios a la imagen*
Se agrega o elimina un registro de la tabla *UsersOfImage*

6.5 *Crear o modificar figura*
Al recibir una petición que modifique una figura se agrega o actualiza el registro en la tabla *figuresOfImage*

7 *Enviar mensajes*
Al recibir una petición de envío de mensajes se concatena el usuario emisor al inicio del mensaje y se envía a los usuarios indicados como receptores del mismo.

El proceso de aprobación de los elementos (imagen o sección) se realiza de la siguiente forma: si el usuario cuyo rol es *aprobador* acepta dicho elemento, se altera una bandera en la tabla *usersOfImage*. Posteriormente el controlador comprueba si todos los usuarios que tienen el rol de *aprobador* han aceptado dicho elemento y se procede a bloquear la edición de éste. En caso de que se modifique la información después de una aprobación, se notifica a los usuarios que revisen los cambios del elemento y se elimina dicha aprobación de la tabla *usersOfImage*.

Capítulo 5
Pruebas de *Misho*

En este capítulo se divide en dos secciones, la sección 5.1 presenta las pruebas realizadas al entorno *Misho* para comprobar el funcionamiento de las partes del sistema y la sección **??** presenta el análisis de resultados, en comparación con los trabajos relacionados presentados en el capítulo de estado del arte.

5.1. Pruebas de *Misho*

Para las pruebas referentes al funcionamiento de *Misho* se utilizaron tres computadoras con distintos sistemas operativos de escritorio (Ubuntu Linux 10.04, Windows 7 y Mac OS X 10.6.8), en las cuales se instaló el cliente de *Misho*. El servidor se instaló en la computadora con Windows. En cuanto a la red, se utilizó una red local con direcciones IP estáticas en cada computadora.

Las operaciones que fueron probadas con tres usuarios fueron:

1 Registro de usuario.

2 Inicio de sesión.

3 Creación y configuración de grupos de trabajo.

4 Edición cooperativo de texto.

- **4.1** Creación de documentos.
- **4.2** Creación de secciones.
- **4.3** Apertura de un documento.
- **4.4** Cierre de un documento.

4.5 Invitación de usuarios a cada sección con los diferentes roles de usuarios (*editor*, *revisor* y *aprobador*).

4.6 Edición de una sección del documento.

4.7 Sincronización de cambios (síncrona, asíncrona y por número de transacciones en palabra o párrafo).

4.8 Notificación de candados de edición.

4.9 Referencia mediante *Deixis* desde el espacio de comunicación al editor cooperativo de texto.

4.10 Consenso de aprobación de una sección.

4.11 Memorización documento.

4.12 Exportación documento.

5 Pizarrón cooperativo

5.1 Creación de imágenes.

5.2 Invitación de usuarios a la modificación de cada imagen con diferentes roles de usuario (*editor*, *revisor* y *aprobador*).

5.3 Edición de una imagen.

5.4 Sincronización de cambios (síncrono y asíncrono).

5.5 Notificación de candados de edición.

5.6 Referencia mendiante *Dexis* desde el espacio de comunicación al pizarrón cooperativo.

5.7 Consenso de aprobación de una imagen.

5.8 Apertura y memorización de una imagen.

5.9 Exportación de una imagen.

6 Sala de mensajería instantánea.

6.1 Conversación general entre todos los usuarios.

6.2 Conversación entre un par de usuarios.

A continuación, se explica la forma en que se realizaron las pruebas antes mencionadas.

Estas pruebas se realizaron con tres usuarios: Laura, Estela y Ernesto. Cada uno de los usuarios se registró e inicio sesión en el cliente correspondiente (véase figura 5.1). Se verificó que el funcionamiento de estas opciones fuera el correcto, así como los cambios que *Misho* realiza en la base de datos del servidor. La figura 5.2 presenta: a) el inicio de sesión del usuario Ernesto, b) la notificación de error de autentificación y c) la ventana de *captcha*.

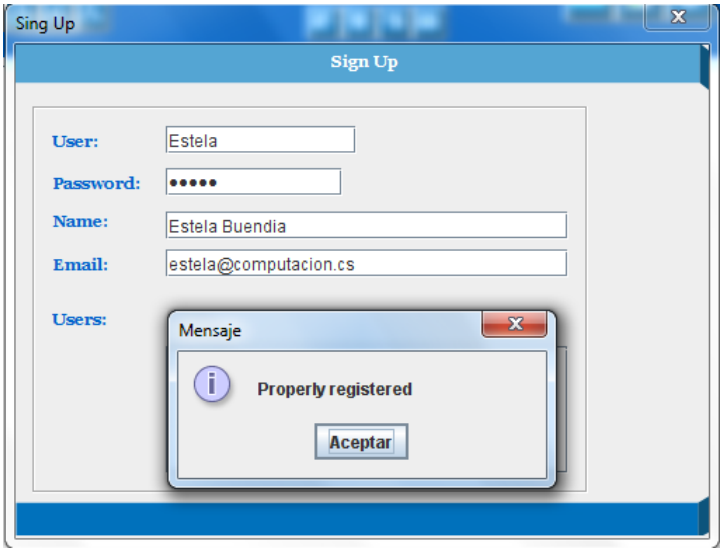
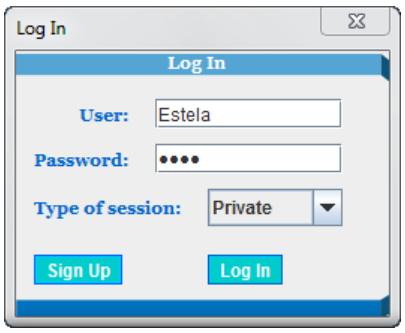
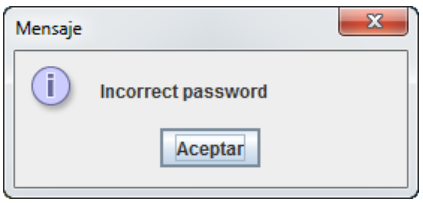


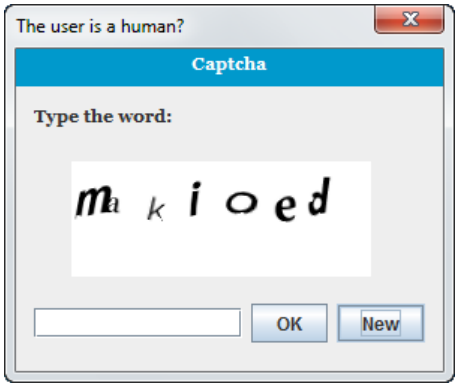
Figura 5.1: Cuadro de diálogo para el registro de usuarios



a) Inicio de sesión



b) Mensaje de error



c) *Captcha*

Figura 5.2: Cuadros de diálogo correspondientes al inicio de sesión

Una vez establecida la autentificación del usuario, por medio del inicio de sesión, se comenzó con las pruebas de: 1) creación y configuración de grupos de trabajo, 2) funcionamiento del editor cooperativo de texto, 3) funcionamiento del pizarrón cooperativo, y finalmente, 4) actividad en la sala de mensajería instantánea.

5.1.1. Creación y configuración de grupos de trabajo

Para crear y configurar los grupos de trabajo, el usuario debe acceder al menú *Workteam* y a la opción con el mismo nombre, la cual despliega el cuadro de diálogo de creación y configuración de grupos de trabajo. En este cuadro de diálogo, el usuario Laura creó los grupos de trabajo *Teoría* e *Desarollo*. También, el usuario Laura invitó al usuario Estela a participar en el grupo de *Teoría* y al usuario Ernesto al grupo de *Desarrollo*. En la figura 5.3 se presenta el cuadro de diálogo con la configuración de ambos grupos de trabajo, tomando como ejemplo el grupo de trabajo de *Teoría*.

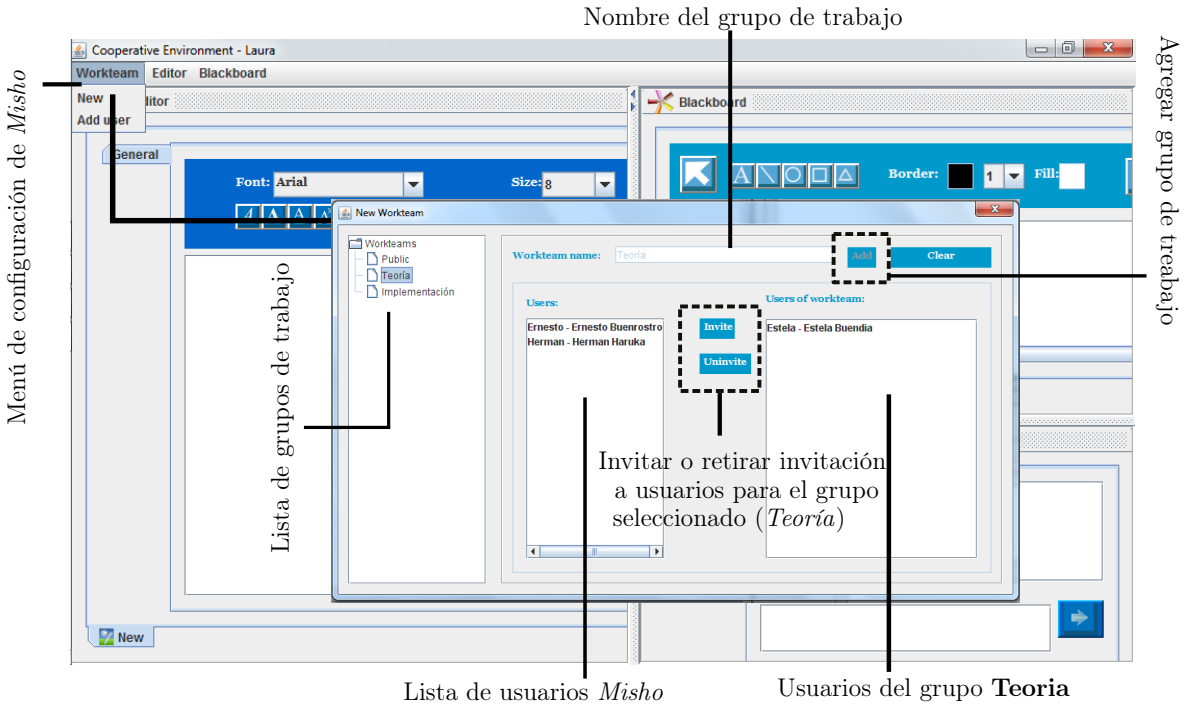


Figura 5.3: Cuadro de diálogo para la creación y configuración de grupos de trabajo

5.1.2. Editor cooperativo de texto

Cuando un usuario inicia sesión en su respectivo cliente *Misho*, se presenta en el editor cooperativo de texto un nuevo documento con una primera sección en la que el usuario puede comenzar a escribir. El contenido del nuevo documento se almacena localmente en la aplicación cliente, mientras el usuario no guarde explícitamente el documento. Cuando el usuario guarda el documento, éste se envía al servidor, que agrega el documento a la

lista de documentos que se pueden trabajar cooperativamente.

Para probar las opciones del editor cooperativo de texto anteriormente descritas, se pensó en utilizar como ejemplo la edición de un artículo, donde cada sección del mismo es presentada como un documento que contiene las subsecciones de la misma.

Misho permite tener un solo documento nuevo o sin guardar, pero permite abrir varios documentos previamente guardados, por lo que, para crear los cinco capítulos (*introducción, estado del arte, análisis y diseño, desarrollo* y *conclusiones*) que forman a está tesis, fue necesario guardar el documento nuevo con el nombre del capítulo correspondiente y seleccionar el grupo de trabajo al que pertenece. Mediante este proceso se probaron las opciones: crear un documento, guardar un documento y abrir un documento. También se hizo la prueba del proceso de invitación de usuarios inscritos al grupo de trabajo al que pertenece el documento.

Para poder describir las opciones que el usuario puede realizar en la edición de un documento, se tomó como ejemplo el documento que representa al capítulo *introducción*. En éste documento se crearon las secciones: *contexto de investigación, antecedentes, planteamiento del problema, objetivos del proyecto y organización del documento*. Las secciones fueron creadas por el usuario Laura por lo que sólo él puede visualizar y editar la información. Hasta que él invite a los usuarios que pertenecen al grupo de trabajo del documento puede trabajarse en dicha sección de forma cooperativa. También se hizo la prueba del proceso de la invitación de usuarios (inscritos al documento) a una sección

Se utilizó la sección *planteamiento del problema* para presentar las pruebas de edición del texto de una sección un documento. Las pruebas que se hicieron en dicha sección son: 1) edición de texto, 2) cambio de estilo en la fuente del texto, 3) sincronización síncrona, 4) sincronización asíncrona, 5) sincronización por palabra, 6) sincronización por párrafo, 7) candados de edición, 8) referencia mediante *deixis*, y finalmente, 9) aprobación de la sección del documento.

A los tres usuarios se les dió una copia del texto de la sección para que lo transcribieran conjuntamente en la sección, utilizando *Misho*, donde dicha sección contiene cambios de estilos en la fuente del texto. Para verificar el funcionamiento de *Misho*, se pidió a los usuarios que intercambiaran el tipo de sincronización, entre asíncrono y síncrono con las diferentes granularidades posibles (letra, palabra o párrafo), durante la edición del texto. La actualización de los cambios en el contenido de la sección del documento fue verificado de esta manera para los cuatro tipos de sincronización. También se observaron las ventajas y desventajas (las cuales se describen a continuación) de usar la técnica implementada para el control de concurrencia.

Las principales ventajas observadas que ofrece el uso de candados en el editor cooperativo de texto en *Misho* son: 1) cada usuario conoce los fragmentos de texto que están sien-

do editados por los demás colaboradores, proporcionando así una percepción que se está trabajando en grupo y 2) la notificación del nombre del usuario que tiene el permiso de edición de dicha palabra.

Las desventajas de usar candados en la edición cooperativa de texto se presentan en dos casos: a) cuando se está utilizando una sincronización de tipo síncrona por párrafo o sincronización de tipo asíncrona y b) cuando se edita la última o penúltima palabra del párrafo. Para el caso (a) los problemas que se presentan son: 1) el tiempo en el que permanece activo el candado para el conjunto palabra puede llegar a ser considerable y 2) cuando se solicita el permiso de edición de una palabra previamente escrita, ya que este proceso actualiza el valor de dicha palabra y pudo haber cambiado mientras uno estaba realizando los cambios de forma local, lo que genera confusión a los usuarios. Para el caso (b) el problema que se presenta es que sólo un usuario pueda editar a la vez el final de un párrafo.

En las figuras 5.4 y 5.5 se presentan las vistas del editor cooperativo de texto con la que interactúan los usuarios Estela y Ernesto. En estas figuras se observa la existencia de los candados de edición en los párrafos uno y tres, ademas de una referencia mediante *deixis* a un fragmento de texto del párrafo dos.

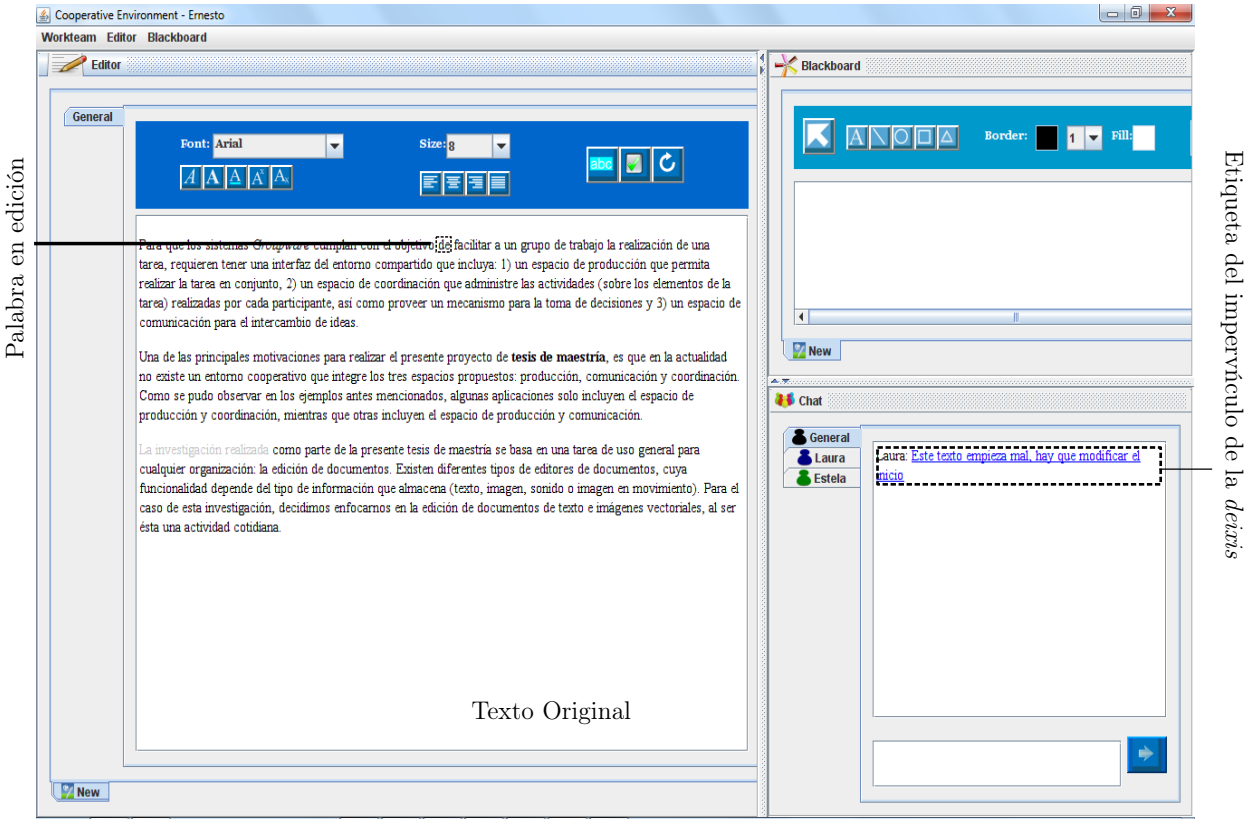


Figura 5.4: Pruebas realizadas al editor cooperativo de texto en el usuario Ernesto

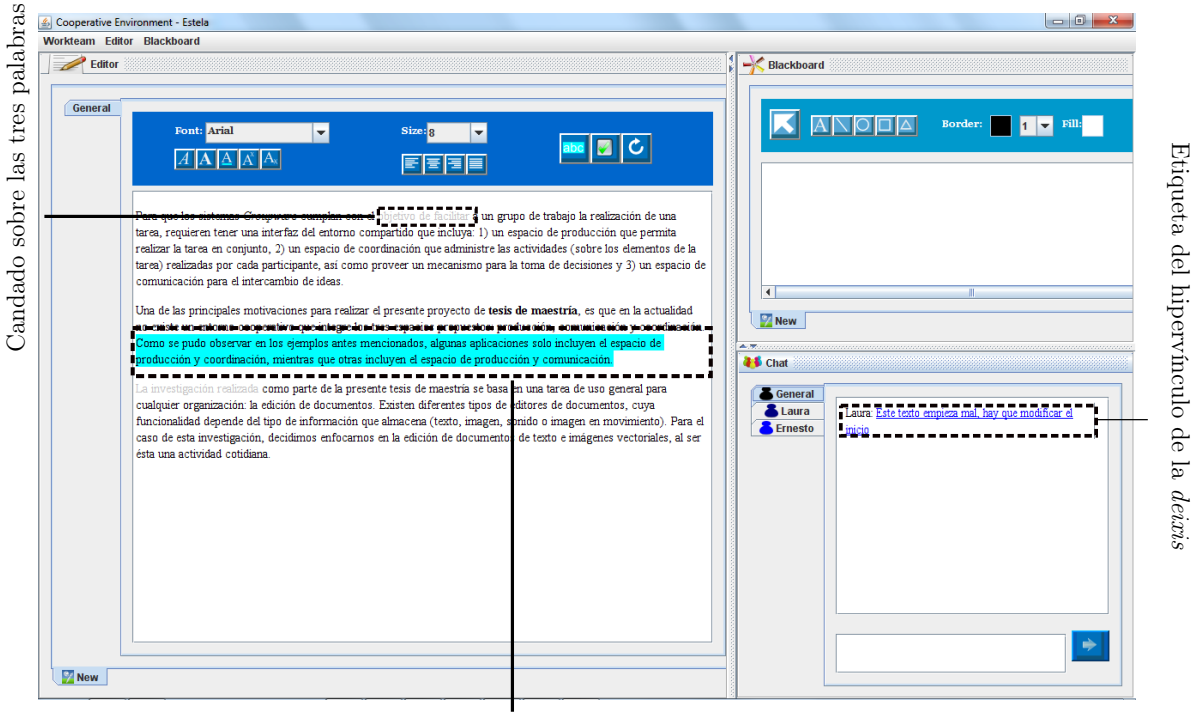


Figura 5.5: Pruebas realizadas al editor cooperativo de texto en el usuario Estela

Para verificar el proceso de aprobación de secciones, se asignó a los usuarios Ernesto y Estela el rol de *aprobador* y se solicitó que cada uno aprobara la sección del documento en las siguientes situaciones: a) sólo el usuario Ernesto aprobó la sección del documento, posteriormente el usuario Laura realizó modificaciones en el texto, esto envío una notificación al usuario Ernesto que se ha modificado dicha sección y que por consiguiente requiere ser aprobada nuevamente; b) el usuario Estela aprobó la sección del documento la cual fue posteriormente aprobada por el usuario Ernesto. Como no hubo ninguna modificación del texto y las dos aprobaciones necesarias se realizaron, se retiró el permiso de edición a todos los usuarios que tienen acceso a dicha sección.

Para verificar que los permisos de usuarios sobre cada sección funcionaran, se configuró la sección *contexto de investigación* como privada para el uso exclusivo del usuario Laura. En la sección *antecedentes* los usuarios tuvieron diferentes roles, e.g. el usuario Laura tuvo el rol de *editor* y el usuario Estela tuvo el rol de *revisor*. Para ambos casos, los procesos funcionaron como se esperaba. El único problema que se presentó en la sección de *antecedentes* fue que el revisor solo puede hacer comentarios a través del proceso de *deixis*, lo que obliga a que estos comentarios se realicen cuando los usuarios se encuentren en línea.

5.1.3. Pizarrón cooperativo

Cuando un usuario inicia una sesión en el cliente *Misho*, se presenta en el pizarrón cooperativo una nueva imagen. De forma similar a la creación de un nuevo documento, mientras el usuario no guarde la imagen, su contenido se almacena de forma local en el cliente. Cuando el usuario guarda la imagen, ésta se envía al servidor y pasa a formar parte de los imágenes que se pueden trabajar cooperativamente.

Para probar las opciones del pizarrón cooperativo anteriormente descritas, se pensó en utilizar un bosquejo de una casa cuadrada con techo rectangular y ventanas redondas (véase figura **??**).

Misho permite tener una sola imagen nueva o sin guardar, pero permite abrir varias imágenes previamente guardadas, por lo que el usuario Estela creó la imagen casa e invitó a los demás colaboradores para su edición. Con este proceso se probaron las opciones: crear una imagen, guardar una imagen, abrir una imagen e invitar usuarios a la edición de una imagen.

Se pidió a los usuarios que intercambiaran durante la edición de la imagen el tipo de sincronización, entre sincronización síncrona y asíncrona, para verificar su funcionamiento. La actualización de los cambios en el contenido de las figuras fue verificado de esta manera para los cuatro tipos de sincronización. También se observaron las ventajas y desventajas de usar la técnica implementada para el control de concurrencia, dichas ventajas y desventajas se describen a continuación.

Las principales ventajas observadas que ofrece el uso de candados en el pizarrón cooperativo en *Misho* son: 1) cada usuario conoce las figuras de la imagen que están siendo editadas por los demás colaboradores, proporcionando así una percepción de que se está trabajando en grupo y 2) la notificación del nombre del usuario que tiene el permiso de edición de dicha figura.

Las desventajas al usar candados en las figuras se presentan cuando se está utilizando un sincronización de tipo asíncrona debido a que el tiempo en el que permanece activo el candado sobre la imagen puede llegar a ser considerable. Otra desventaja que tiene el pizarrón cooperativo es que cada figura se almacena por capas dentro de la imagen, entonces si una figura se encima con otra, solo se podrá tener acceso a la última figura ingresada. Por lo tanto, si desea modificar una figura que se encuentra en una capa inferior, primero es necesario mover las figuras de las capas superiores.

Para verificar que el proceso de *deixis* funcionara correctamente, el usuario Estela seleccionó un fragmento del texto y solicitó que se compartiera a los colaboradores (figura selec-

cionada al momento de compartir). Como se explicó anteriormente, esta opción despliega un cuadro de diálogo que solicita ingresar un nombre para la etiqueta del hipervínculo que se comparte por la sala de conversación; después de que el usuario ingresa el nombre de la etiqueta, se genera automáticamente un hipervínculo en el área de mensaje de dicha sala de conversación. Para poder compartir el hipervínculo, es necesario que el usuario envíe el mensaje, ya que no se envía de forma automática. Cuando el mensaje que contiene el hipervínculo es desplegado en el área de visualización de mensajes, cada usuario puede dar unclick izquierdo de ratón sobre dicho hipervínculo para conocer la parte de la figura que se compartió. En este proceso se presentaron dos problemas: 1) solo se puede hacer *deixis* sobre una figura y 2) el usuario que comparte requiere indicar en el mensaje a qué imagen pertenece dicho hipervínculo.

En las figuras 5.6 y 5.7 se presentan la vista del pizarrón cooperativo con la que interactúan los usuarios Estela y Ernesto. En estas figuras se observa la existencia de un candado en el circulo izquierdo que está seleccionado por el usuario Estela, cambiando las propiedades de su borde y relleno a un color gris y que se generó una referencia mendiante *deixis* sobre el techo (formado por un triángulo) de la casa, el cual parpadea su borde con un color cyan.

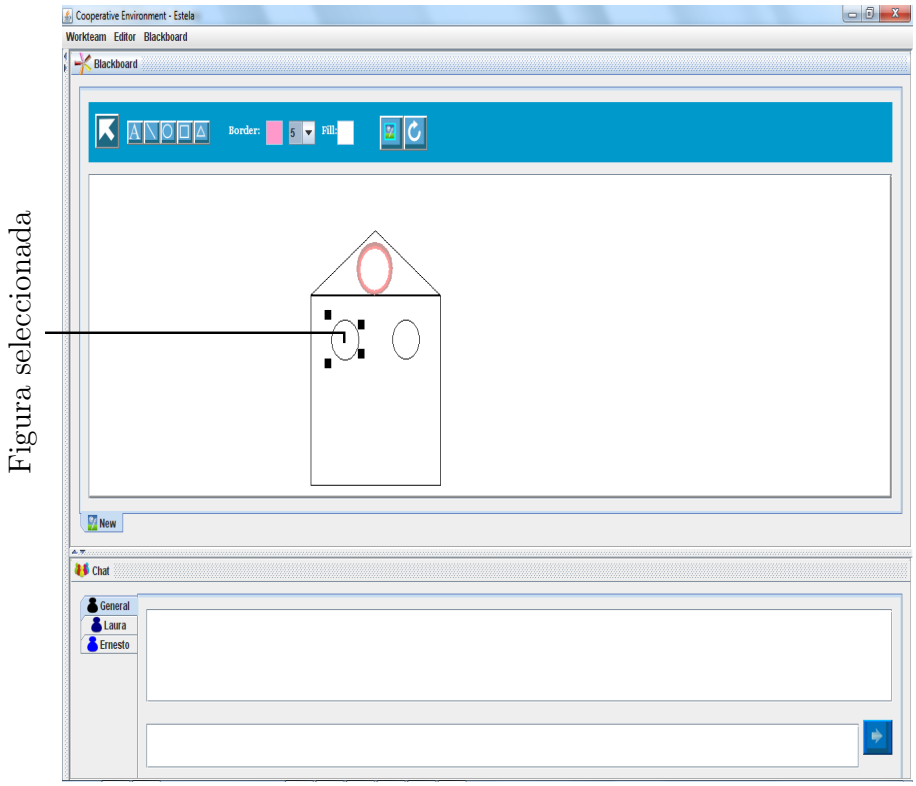


Figura 5.6: Pruebas realizadas al editor cooperativo de texto del usuario Estela

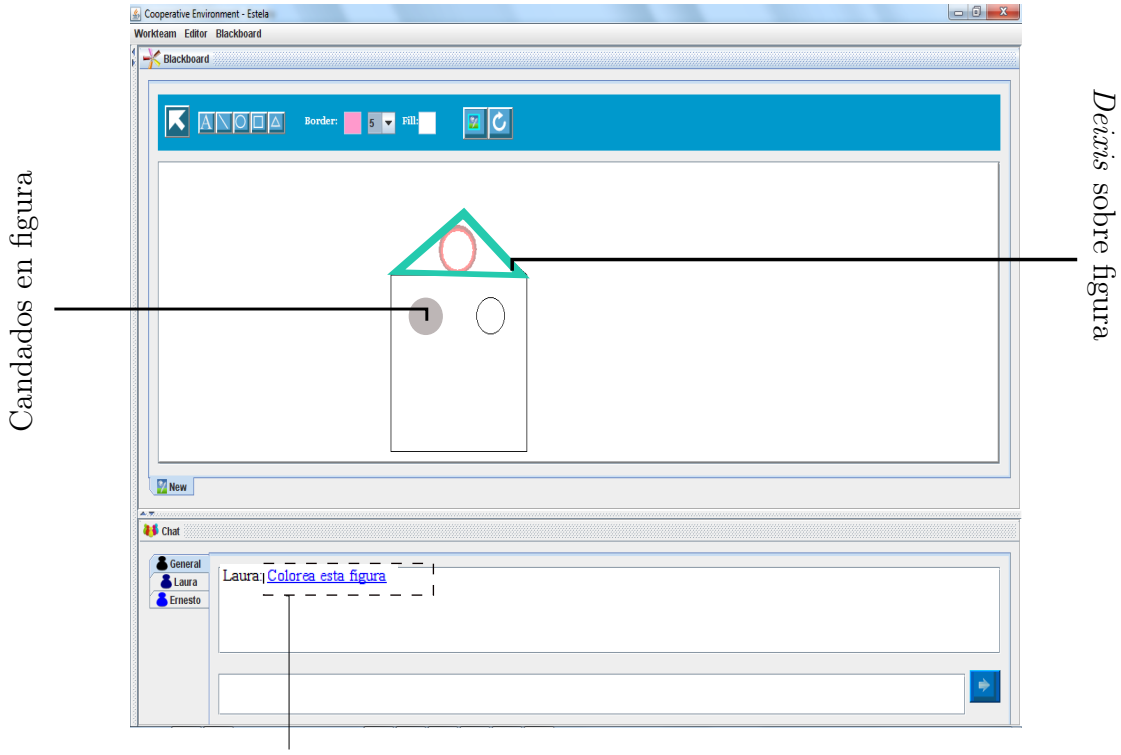


Figura 5.7: Pruebas realizadas al editor cooperativo de texto del usuario Ernesto

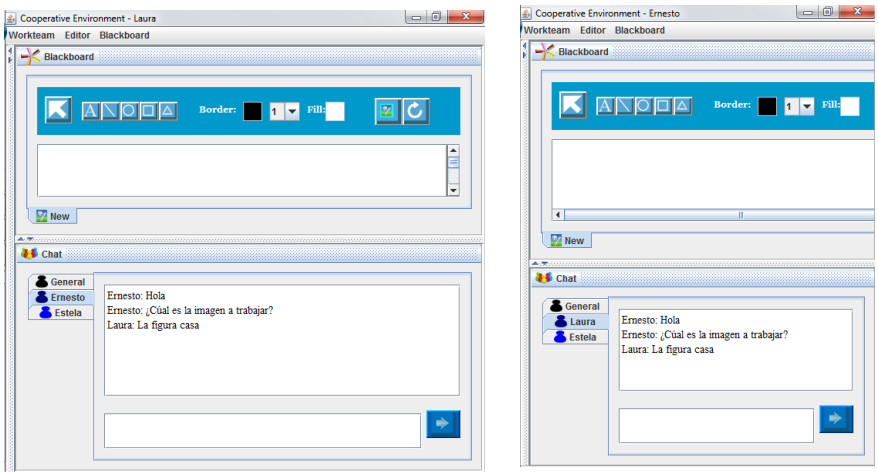
Para verificar el proceso de aprobación de una imagen, se asignó al usuario Laura el rol de *aprobador*, misma que aprobó dicha imagen suprimiendo así el permiso de edición a todos los usuarios que tienen acceso a está imagen.

La verificación de permisos de usuarios sobre cada imagen se probó creando una nueva imagen de un carro y dejando que sólo el usuario Estela pudiera editarla, proceso que realizó correctamente.

5.1.4. Sala de mensajería instantánea

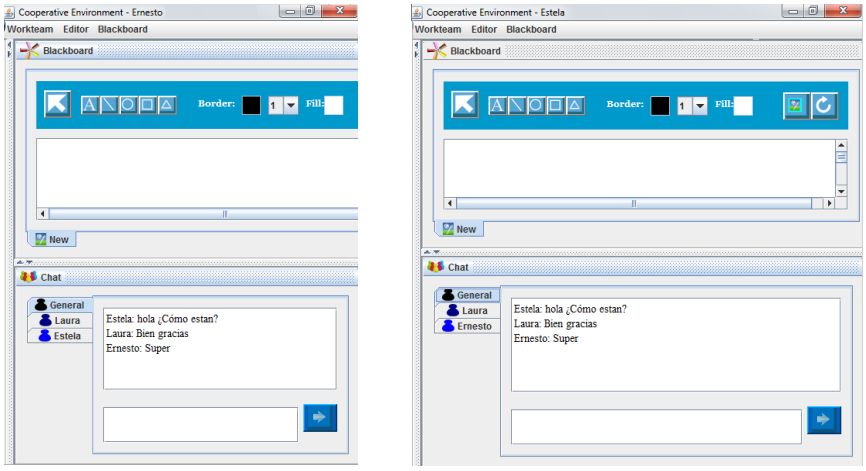
Las pruebas que se hicieron en la sala mensajería instantánea fueron dos: 1) que cuando se enviara un mensaje de forma general éste apareciera en dicha conversación de los usuarios conectados en ese momento y 2) que cuando se envía un mensaje entre un par de usuarios, se desplegara en la conversación correspondiente y solo ellos pudieran tener acceso a dicho mensaje. Ambas pruebas fueron satisfactorias y se presentan en las figuras 5.8 y 5.9.

La figura 5.8 presenta la conversación que mantienen los usuarios Ernesto y Laura, la cual es una conversación privada entre un par de usuarios. Mientras que la figura 5.9 presenta la conversación que se mantiene con todos los colaboradores que están en la sesión en *Misho*.



Conversación entre Ernesto y Laura

Figura 5.8: Pruebas realizadas al editor cooperativo de texto -Conversación privada



Conversación común entre los usuarios de *Misho*

Figura 5.9: Pruebas realizadas al editor cooperativo de texto -Conversación común

5.2. Análisis de resultados

A continuación se presenta una comparación de *Misho* con respecto a los los sistemas estudiados en el capítulo 6, utilizando los mismos criterios de comparación descritos en

ese capítulo.

En *Misho* se incluyó: 1) un proceso de comunicación básico, utilizando mensajería instantánea pero, al ser un sistema modular es posible que en trabajo futuro se pueda incluir un proceso de conferencia electrónica; 2) un proceso de toma de decisiones que permite aprobar los documentos que se editan; 3) un proceso de administración del documento, mediante la planificación anticipada del documento, dividiendo la información en secciones y finalmente 4) la edición cooperativa de documentos de texto y de imágenes vectoriales. En comparación con los sistemas estudiados, *Misho* ofrece el soporte para la toma de decisiones y el mecanismo que da soporte a la *deixis* sobre los espacios de edición, lo cual lo distingue de los demás. La tabla 5.1 presenta dicha comparación con respecto al criterio de funcionalidad.

		Sistemas *Groupware*											
Criterios		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord	***Misho***
Funcionalidad	Mensajería instantánea	-	-	-	X	-	X	X	X	-	-	-	X
	Conferencias y reuniones electrónicas	-	-	-	-	-	-	X	X	-	-	-	-
	Sistema de soporte a la toma de decisiones	-	-	-	-	-	-	-	-	-	-	-	X
	Administración de documentos	X	X	-	-	-	-	-	X	-	-	-	X
	Edición cooperativa	-	-	X	X	X	X	X	X	X	X	X	X

Tabla 5.1: Comparación de *Misho* con los sistemas estudiados con respecto al criterio de funcionalidad

Con respecto al criterio arquitectónico, en *Misho* se utilizó una arquitectura centralizada también conocida como cliente-servidor, utilizando la técnica *push*[1] para realizar las operaciones que realizan los clientes pertenecientes al entorno cooperativo. También cada cliente mantiene una replica local de la información utilizada por el usuario en cada sesión. *Misho* sigue la tendencia actual con respecto a la arquitectura de los sistemas estudiados, como se presenta en la tabla 5.2.

		Sistemas *Groupware*											
Criterios		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord	***Misho***
Arquitectónico	Centralizada	X	-	-	X	X	X	-	-	X	X	X	X
	Replicado	-	-	-	-	-	-	-	-	-	-	X	-
	Híbrido	-	X	X	-	-	-	X	X	-	-	-	-

Tabla 5.2: Comparación de *Misho* con los sistemas estudiados con respecto al criterio de la arquitectura

Misho tomó en consideración para su diseño los dos de los tres criterios del enfoque: 1) con respecto al usuario, *Misho* ofrece una interfaz de usuario que es amigable y mecanismos que facilitan la interacción entre los miembros del equipo de trabajo; y 2) con respecto a la tarea, *Misho* proporciona mecanismos que permiten realizar la tarea conjuntamente dando una consciencia de que se está trabajando en grupo. La tabla 5.3 presenta la comparación realizada con respecto al criterio del enfoque.

		Sistemas *Groupware*											
Criterios		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord	***Misho***
Enfoque	Centrado al usuario	-	-	-	-	-	-	X	X	-	-	-	X
	Centrado a la tarea	X	X	X	X	X	X	X	X	X	X	X	X
	Centrado al espacio de trabajo	-	-	-	-	-	-	-	-	-	-	-	-

Tabla 5.3: Comparación de *Misho* con los sistemas estudiados con respecto al criterio de enfoque

Una ventaja que ofrece *Misho* en comparación a los editores estudiados, es la posibilidad de que el usuario elija el tipo de sincronización (síncrona o asíncrona) por cada imagen o sección del documento que esté editando. Mientras que los demás editores solo ofrecen un tipo de sincronización. La tabla 5.4 presenta la comparación realizada entre de *Misho*

[1]La técnica *push* describe un estilo de comunicación donde la petición de una transacción se origina en el servidor.

con los sistemas estudiados con respecto al criterio del tiempo.

Criterios		Sistemas *Groupware*											
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord	***Misho***
Con base en el tiempo	En tiempo real	X	X	X	-	-	X	-	-	X	X	-	X
	Asíncrono	-	-	-	X	X	X	-	X	X	-	-	X
	Mixto	-	-	-	-	-	-	-	-	-	-	-	X

Tabla 5.4: Comparación de *Misho* con los sistemas estudiado con respecto al criterio del tiempo

Con el fin de ofrecer un sistema multi-plataforma, *Misho* se desarrolló utilizando el lenguaje de programación de Java. También se utilizó el framework de Kryo, el cual permite la clonación de información mediante la serialización de objetos de Java en una red de computadoras y es expansible a dispositivos móviles de Android, por lo cual es una ventaja a comparación de los editores actuales. La tabla 5.5 presenta la comparación de *Misho* con los sistemas estudiados con respecto al criterio de la plataforma.

Criterios		Sistemas *Groupware*											
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord	***Misho***
Plataforma	Sistema Operativo	X	-	X	X	-	-	-	-	X	-	X	-
	Navegador Web	-	-	-	-	-	-	-	X	X	-	-	-
	Multi-plataforma	-	X	-	-	X	X	-	-	-	X	-	X
	Móvil	-	-	-	-	-	-	-	-	-	-	-	-

Tabla 5.5: Comparación de *Misho* con los sistemas estudiado con respecto al criterio de la plataforma

La participación del usuario en *Misho* es media, debido a que requiere establecer una configuración de la forma en que desea trabajar cada documento o imagen, sin embargo, *Misho* realiza procedimientos de forma automática, como son los mecanismos de candados y la sincronización de tipo síncrona. En comparación con los otros sistemas cooperativos, dicha participación del usuario tiene una razón de ser, debido a que *Misho* ofrece mecanismos que permiten la planificación y estructuración del documento, así como limitar los accesos a la información, mismos que deben ser asignados por un miembro del grupo de trabajo y no pueden ser asignados de forma automática por *Misho* (véase la tabla 5.6).

Misho ofrece un control de concurrencia con base en candados, que en comparación a los sistemas estudiados, evita los problemas de coherencia en la información generados por el

Criterios		Sistemas *Groupware*											
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord	***Misho***
Participación del usuario	Alto	X	X	-	X	-	-	-	-	-	-	-	-
	Medio	-	-	X	-	X	X	-	-	X	X	X	X
	Bajo	-	-	-	-	-	-	-	X	X	-	-	-

Tabla 5.6: Comparación de *Misho* con los sistemas estudiado con respecto al criterio de la participación del usuario

acceso a los recursos compartidos. Además, ofrece un bloqueo por granularidad pequeña (tres palabras para el editor y una figura), permitiendo así la edición de varios colaboradores en un mismo documento o imagen a la vez. La tabla 5.7 presenta la comparación de *Misho* con los sistemas estudiados con respecto al criterio del control de concurrencia.

Criterios		Sistemas *Groupware*											
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord	***Misho***
Control de concurrencia	Serialización	X	X	-	-	-	-	-	-	-	-	-	-
	Bloqueo exclusivo	-	-	X	-	-	X	X	-	-	-	-	X
	Algoritmo específico	-	-	-	-	-	-	-	-	-	-	-	-

Tabla 5.7: Comparación de *Misho* con los sistemas estudiados con respecto al criterio de control de concurrencia

La notificación de cambios se realiza en *Misho* al resaltar el texto o la figura informando así a los colaboradores cuáles se mantienen en uso. También se utiliza en el proceso de *deixis* para indicar qué texto o figura ha compartido otro usuario, lo cual es útil cuando la interacción de los usuarios es al mismo tiempo, sin embargo el proceso de *deixis* está limitado a la interacción asíncrona, ya que no se pueden hacer discusiones. La tabla 5.8 presenta la comparación con base al criterio de notificación de cambios.

Criterios		Sistemas *Groupware*											
		iFolder	Dropbox	SubEthaEdit	SharePoint	Abiword	Gobby	ShowDocuments	GoogleDocs	CollabEdit	CoopDraw	CodoxWord	***Misho***
Notificación de cambios	Notas o comentarios	-	-	X	X	-	-	X	X	X	-	X	-
	Resaltado de texto	-	-	-	-	X	X	-	-	-	-	-	X

Tabla 5.8: Comparación de *Misho* con los sistemas estudiados con respecto al criterio a la notificación de cambios

Capítulo 6
Conclusiones y Trabajos Futuros

Este capítulo se divide en tres secciones. La sección 6.1 hace una recapitulación de la problemática atacada por la propuesta de *Misho*. La sección 6.2 describe las conclusiones derivadas de la presente tesis de maestría. Finalmente, la sección 6.3 describe las extensiones y el trabajo futuro que pueden realizarse sobre el entorno cooperativo desarrollado, con el fin de ampliar y mejorar su funcionalidad.

6.1. Recapitulación

En los últimos años han aparecido algunas aplicaciones que implementan funcionalidades de los sistemas *Groupware* como son: la administración de archivos, la edición cooperativa de documentos, la sala de conversación, el soporte para la toma de decisiones, el manejo de usuarios y roles, etc. Sin embargo, dichas aplicaciones dan prioridad a uno o dos de los espacios funcionales de *Groupware*, dejando la responsabilidad de los espacios faltantes a los miembros del grupo de trabajo.

Si se integran los tres espacios en un solo entorno, se puede aumentar la eficacia de los colaboradores al realizar una tarea común, proporcionando un área de trabajo que de la percepción que se está trabajando en grupo, un mecanismo por el cuales pueden comunicar los colaboradores entre sí y, a su vez, proporcionar herramientas que faciliten la coordinación y administración de las actividades de los miembros del grupo de trabajo.

La edición cooperativa de documentos es una actividad donde varias personas trabajan conjuntamente con una meta común. Los participantes pueden ejercer diferentes roles durante la edición de un documento y requieren un control sobre: 1) las actividades (planeación, escritura, redacción y revisión), 2) la estructura del documento y 3) el acceso a las secciones del mismo para su edición.

Se estudiaron las necesidades que presentan los usuarios al editar cooperativamente un documento, con el fin de proporcionarle las herramientas necesarias que estos requieren. Dentro de las necesidades que se detectaron se encuentran: 1) la edición de documentos también requiere la edición de imágenes que complementen el texto del documento, 2) los colaboradores requieren conocer los cambios realizados por los demás, con el fin de evitar problemas de concurrencias en la edición del documento, 3) las salas de conversación presentan una limitante para los colaboradores al no permitir los mismos gestos que se pueden utilizar en una interacción presencial, 4) en ocasiones, los colaboradores que editan el documento no requieren tener acceso a toda la información del mismo, 5) se requiere una estructura jerárquica para la planificación del documento, 6) los participantes ejercen diferentes roles durante la edición del documento y 7) se requiere un soporte para la toma de decisiones, en la cual los colaboradores puedan decidir cuando se a terminado dicho documento.

6.2. Conclusiones

Se logró integrar los espacios funcionales de *Groupware* en un sistema que permite la edición cooperativa de documentos e imágenes, recreando la interacción física que tienen los usuarios en la realización de este tipo de tareas. *Misho* permite romper con los limites que se generan al interactuar por medio de una red de computadoras, ya que se implementó un mecanismo de notificación de las actividades realizadas por cada usuario sobre los recursos compartidos. Así mismo, se integró un proceso de *deixis* al espacio de comunicación, con el fin de imitar el señalamiento físico de una parte del espacio de producción.

Debido al hecho de que los sistemas *Groupware* están basados en sistemas distribuidos, se optó por diseñar una aplicación cliente-servidor, la cual utiliza un medio de comunicación en red que permita su portabilidad a diferentes plataformas de escritorio e incluso dispositivos móviles, aunque *Misho* se enfocó las plataformas de escritorio Linux, Mac OS X y Windows.

Al utilizar el patrón de desarrollo Modelo-Vista-Controlador se garantiza la flexibilidad a los cambios y la escalabilidad de *Misho*, propiedades que se obtuvieron al desarrollar módulos independientes de cada espacio, i.e. tanto el editor cooperativo de texto, el pizarrón y la sala de mensajería son independientes entre sí, de esta forma, cualquier cambio en realizado ellas no afecta desempeño de los demás. Ademas *Misho*ofrece la posibilidad de agregar más funcionalidades al entorno.

Esta investigación sobre la edición cooperativa de documentos explora las opciones que pueden ser ofrecidas al grupo de trabajo para cada espacio funcional, con la finalidad de aumentar la conciencia del trabajo en equipo y mantener la coherencia en la información durante la tarea realizada, ofreciendo una administración centralizada de la información global de *Misho*.

Se exploraron las técnicas y tecnologías que permiten dar solución a los problemas existentes en los grupos de trabajo, en cuanto a la sincronización de cambios, el control de concurrencia en los recursos compartidos y las tecnologías de comunicación en red, con el fin de implementar las alternativas que ofrezcan mayor funcionalidad a los colaboradores durante la edición de un documento. Además se crearon mecanismos no explorados en los editores cooperativos actuales, como son el uso de hipenvínculos para el soporte de la *deixis* y la toma de decisiones para aprobar la información del documento.

Misho da la oportunidad a cada usuario de configurar las actualizaciones de la información de cada sección del documento o de la imagen, de acuerdo a sus necesidades. Asimismo, *Misho* establece un proceso de control de concurrencia a través de candados, los cuales permiten notificar a los usuarios qué fragmentos están siendo utilizados por demás usuarios, y de esta forma, evitar incoherencias en la información al tener recursos compartidos.

Misho administra a los usuarios, grupos de trabajo, documentos e imágenes con base en estructuras jerárquicas, con las cuales se facilita el control de permisos de acceso a la información. Cada usuario maneja un rol en cada sección o imagen.

6.3. Trabajo Futuro

Dentro del trabajo futuro que se puede implementar en *Misho* está:

- Implementar el modelo vista de la aplicación del cliente de *Misho* para que sea soportado en dispositivos móviles con plataforma Android.
- Modificar la estructura híbrida basada en cliente-servidor por una de tipo peer-to-peer o mediante servicios web para comprobar cuál es la mejor arquitectura para su correcto funcionamiento.
- Enviar notificaciones de invitaciones a documentos o imágenes al correo electrónico de los miembros del grupo de trabajo; tal como lo hace la aplicación Dropbox, para que el usuario sea notificado y decida si quiere pertenecer o no.
- Cambiar la visualización del cliente de *Misho* de una sola ventana a varias ventanas, para que cada colaborado elija la posición de los elementos que lo componen (editor cooperativo de texto, pizarrón cooperativo y sala de mensajería instantánea).
- Incluir la opción de crear notas o comentarios dentro del documento o la imagen, como apoyo a los revisores.

Bibliografía

[Cantero et al., 2001] Cantero, M., Rodríguez, J. and Bravo, J., *Sistemas de Interacción Persona-Computador*, Colección Ciencia y Técnica/Ediciones de la Universidad de Castilla-La Mancha Series, Ediciones de la Universidad de Castilla-La Mancha, 2001.

[Coulouris and Dollimore, 1988] Coulouris, G.F. and Dollimore, J., *Distributed systems: concepts and design*, Addison-Wesley Longman Publishing Co., Inc., Boston, MA, USA, 1988.

[Coutaz and Calvary, 2007] Coutaz, J. and Calvary, G., *The Case for User Interface Plasticity, book The Human-Computer Interaction*, chap. 56, Handbook: Fundamentals, Evolving Technologies, and Emerging Applications, 2007.

[Coutaz and Rey, 2002] Coutaz, J. and Rey, G., *Foundations for a theory of contextors*, in CADUI, pp. 13–34, 2002.

[Crowley et al., 2002] Crowley, J.L., Coutaz, J., Rey, G. and Reignier, P., *Perceptual components for context aware computing*, 2002.

[Decouchant et al., 1999] Decouchant, D., Enríquez, A.M.M. and González, E.M., *Alliance web: Cooperative authoring on the www*, in SPIRE/CRIWG, pp. 286–295, 1999.

[Decouchant et al., 2009] Decouchant, D., Escalada-Imaz, G., Martinez Enriquez, A.M., Mendoza, S. and Muhammad, A., *Contextual awareness based communication and coauthoring proximity in the internet*, Expert Syst. Appl., vol. 36, pp. 8391–8406, Pergamon Press, Inc., Tarrytown, NY, USA, Mayo 2009, ISSN 0957-4174.

[Dey and Abownd, 1999] Dey, A.K. and Abownd, G.D., *Towards a better understanding of context and context-awareness*, in Handheld and ubiqitous computing H.W. Gellerson ed., no. 1707 in Lecture Notes in Computer Science, Springer, pp. 304–7, Septiembre 1999.

[Dey and Mankoff, 2005] Dey, A.K. and Mankoff, J., *Designing mediation for context-aware applications*, ACM Trans. Comput.-Hum. Interact., vol. 12, pp. 53–80, ACM, Marzo 2005.

[Dourish and Bly, 1992] Dourish, P. and Bly, S., *Portholes: supporting awareness in a distributed work group*, in Proceedings of the conference on Human Factors in Computing Systems, CHI '92, ACM, pp. 541–547, New York, NY, USA, 1992.

[Ellis et al., 1991] Ellis, C.A., Gibbs, S.J. and Rein, G., *Groupware: some issues and experiences*, ACM, vol. 34, pp. 39–58, ACM, Enero 1991.

[Ellis and Wainer, 1994] Ellis, C.A. and Wainer, J., *A conceptual model of groupware*, in Proceedings of the 1994 ACM conference on Computer Supported Cooperative Work, CSCW '94, ACM, pp. 79–88, 1994.

[Engelbart, 1968] Engelbart, D.G., *The mother of all demos*, 1968.

[Erkens et al., 2002] Erkens, G., Kanselaar, G., Prangsma, M. and Jaspers, J., *Using tools and resources in computer supported collaborative writing*, in Proceedings of the Conference on Computer Support for Collaborative Learning: Foundations for a CSCL Community, CSCL '02, International Society of the Learning Sciences, pp. 389–398, 2002.

[Fraser, 2011] Fraser, N., *Version control workshop.*, in ACM Symposium on Document Engineering M.R.B. Hardy and F.W. Tompa eds., ACM, pp. 267–268, 2011.

[Gamma, 1995] Gamma, E., *Patrones de diseño: elementos de software orientado a objetos reutilizable*, Addison-Wesley professional computing series, Reading, Massachusetts ; Madrid [etc] : Addison-Wesley Publishing Company, 1995.

[Granville and Hickey, 2009] Granville, K.G. and Hickey, T.J., *Collabed: A platform for collaboratizing existing editors*, in Proceedings of the 2009 International Conference on Mobile, Hybrid, and On-line Learning, IEEE Computer Society, pp. 90–96, Washington, DC, USA, 2009.

[Greif, 1988] Greif, I., *Computer-supported cooperative work: a book of readings*, Morgan Kaufmann Publishers Inc., San Francisco, CA, USA, 1988.

[Haake et al., 2010] Haake, J., Hussein, T., Joop, B., Lukosch, S., Veiel, D. and Ziegler, J., *Modeling and exploiting context for adaptive collaboration*, International Journal of Cooperative Information Systems (IJCIS), vol. 19, no. 1-2, pp. 71–120, 2010.

[Hofte and Opendoc, 1997] Hofte and Opendoc, B.O., *Cocodoc: a framework for collaborative compound document editing based on opendoc and corba*, 1997.

[Mitchell, 1996] Mitchell, A., *Communication and Shared Understanding in Collaborative Writing*, Ph.D. thesis, Department of Computer Science, University of Toronto, 1996.

[Noël and Robert, 2004] Noël, S. and Robert, J.M., *Empirical study on collaborative writing: What do co-authors do, use, and like?*, Computer Supported Cooperative Work (CSCW), vol. 13, no. 1, pp. 63–89, Kluwer Academic Publishers, Norwell, MA, USA, Jan. 2004.

[Ramstein, 1993] Ramstein, C., *Coopdraw: a multiagent architecture for a shared graphical editor*, in Proceedings of the 1993 conference of the Centre for Advanced Studies on Collaborative research: distributed computing - Volume 2, CASCON '93, IBM Press, pp. 758–766, 1993.

[Saltiveri, 2005] Saltiveri, T., *Diseño de sistemas interactivos centrados en el usuario*, UOC, Barcelona, 2005, ISBN 8497883209.

[Schilit et al., 1994] Schilit, B., Adams, N. and Want, R., *Context-aware computing applications*, in Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications, WMCSA '94, IEEE Computer Society, pp. 85–90, Washington, DC, USA, 1994.

[Schmidt and Bannon, 1992] Schmidt, K. and Bannon, L., *Taking cscw seriously*, Computer Supported Cooperative Work (CSCW), vol. 1, pp. 7–40, 1992.

[Whalen, 1997] Whalen, T., *Design issues for an adaptive mobile group editor*, 1997.