Centro de Investigación y de Estudios
Avanzados
del Instituto Politécnico Nacional

Unidad Zacatenco

Departamento de Computación

Colaboración cara a cara mediante arreglos de dispositivos móviles

Tesis que presenta

Eric Abraham Vargas Flores

para obtener el Grado de

Maestro en Ciencias

en Computación

Directora de la Tesis

Dr. Sonia Guadalupe Mendoza Chapa

México,D.F. Noviembre 2013

Índice general

Índice de figuras

Capítulo 1

Introducción

1.1. Contexto de la investigación

En estos tiempos, somos testigos de cambios en la tecnología que hacen que cualquier persona pueda beneficiarse de ésta en cierta forma. Así mismo la tecnología es más inmersa conforme pasa el tiempo. Podriamos considerar a la escritura como la primera tecnología de información, ya que podemos representar el lenguaje hablado en símbolos que pueden perdurar por mucho tiempo, guardados en libros y hojas. Ahora esa tecnología es tan natural para nosotros que ni siquiera la catalogamos como tecnología. Nos encontramos en un punto de la historia en el que las computadoras están dejando de ser solo máquinas usadas para llevar a cabo tareas específicas, para algunos grupos de personas, si no que esta pasando lo mismo que paso con la escritura, las computadoras se están transformando en parte esencial de nuestra vida cotidiana porque nos ayudan a ser más eficientes y nos información y servicios que nos apoyan en nuestras actividades.

La visión de que las computadoras estarán tan inmersas en nuestra existencia que no lo notariamos, ha pasado por nuestras cabezas desde hace ya varios años, tanto en novelas de ciencia ficción como en películas, aunque el primer autor en mencionarlo fue Mark Weiser, ya que acuñó el término de *cómputo ubicuo* [Weiser, 1991]. El cómputo ubicuo es la idea de integrar a las computadoras en el mundo y que pasen desapercibidas por las personas, tal como lo hace la escritura hoy en día. Por lo tanto, en un futuro no muy lejano llegaremos a usar las computadoras como accesorios, artefactos que existen en el medio ambiente, con los que haremos las interacciones naturales o actividades que llevamos a cabo diariamente.

Podemos darnos cuenta de que estamos acercandonos a la visión del cómputo ubicuo,

porque en un lapso muy corto de tiempo, hemos visto una creciente proliferación de dispositivos de cómputo. Ahora nuestros dispositivos móviles son más poderosos que las computadoras de la década pasada, y son mucho más pequeños ya que los llevamos a todas partes en nuestros bolsillos. Si esto ha pasado en tan poco tiempo, imaginemos las computadoras de los próximos 10 o 20 años.

Para llevar esto a cabo, se trabajan en diferentes áreas de la computación, para crear investigación y tecnología que nos ayude a implementar el cómputo ubicuo. Una de estas áreas es la Interacción Hombre-Máquina (IHM), donde se estudia todo lo relacionado con las formas en las que las personas interactúan con un sistema de cómputo. La existencia de tantos dispositivos de cómputo, de redes de comunicación que los comunican, y una mayor capacidad de cómputo ofrece nuevos desafíos en en desarrollo de tecnología dentro de la comunidad de IHM [Thevenin and Coutaz, 1999]. Esto es porque interactuamos con los dispositivos de diferentes maneras, existen periféricos como el mouse y teclado, pero ahora también los dispositivos permiten usar el tacto de las personas con solo tocar las pantallas. Pero el contar con una gran gama de dispositivos quiere decir que cada uno cuenta con su plataforma (como el sistema operativo, y la arquitectura del dispositivo) específica, además de un tamaño de pantalla diferente, ya que existen computadoras de escritorio (PC), laptops, dispositivos móviles, tablets, entre otros.

Existen diversos mecanismos para implementar interfaces de usuario en dispositivos que tienen diferentes plataformas. Puede ser algo sencillo como el reducir el tamaño de los componentes gráficos que se muestran en pantalla, así como el cambio de posición de los elementos para que se vean mejor organizados, pero también puede ser algo complejo como el cambiar la interfaz en tiempo de ejecución del programa, tomando en cuenta *variables del contexto*.

El término de *cómputo consciente del contexto* [Schilit et al., 1994], considera importante que el software debe de ser reactivo a la localización del usuario, la gente que hay alrededor, los dispositivos disponibles, así como los cambios que puedan ocurrir en el tiempo. Por lo tanto se consideran variables de contexto, a aspectos como en donde te localizas, con quien estas, y que recursos tienes a tu alcance o se encuentran a tu alrededor. Una característica significativa de este tipo de software es el constante cambio que existe en el ambiente de ejecución. Los dispositivos disponibles para los usuarios, la capacidad de red y conectividad, y el costo de procesamiento, todos pueden cambiar conforme transcurre el tiempo. Al mismo tiempo, un usuario también podría cambiarse de localización, podría unirse o salirse de un grupo de personas o colaboradores, y podría interactuar con diferentes dispositivos.

La consciencia del contexto es importante en el área de IHM, porque como hemos mencionado, la interfaz de usuario puede variar dependiendo de las características físicas del dispositivo, pero tomando en cuenta también las variables de contexto,

entonces tenemos un sistema más complejo, aunque logra que los usuarios tengan una experiencia de uso más enriquecedora y natural, ya que el sistema se adaptará a las acciones que lleva a cabo el usuario.

Existe otra área de conocida como Trabajo Colaborativo Asistido por Computadoras (TCAC) [Greif, 1988], en la cual se estudian las interacciones que tienen las personas en un grupo para así ofrecer un software especializado para que hagan efectivamente sus tareas. A este tipo de software se le conoce como *sistema colaborativo*, ya que toma en cuenta las interacción que existe en un grupo social para llevar a cabo una adaptación en tiempo real. Para dar soporte a un grupo de personas dentro de un sistema es escencial tomar en cuenta algunos aspectos: comunicación, colaboración, y coordinación [Ellis et al., 1991]. El ejemplo más básico de sistema de software que es catalogado como un sistema colaborativo, es un chat, ya que une a un grupo de personas de manera virtual, apra que se comuniquen entre sí, y ésta comunicación está coordinada por el sistema.

El presente trabajo de investigación, tiene como finalidad aportar más conocimiento dentro del área de TCAC, creando un sistema colaborativo que pueda adaptarse a las necesidades del usuario, en su interfaz de usuario. Para llevar a cabo esta adaptación, también es importante conocer sobre la plasticidad de las interfaces gráficas de usuario.

El concepto de *plasticidad* esta inspirado en las propiedades de los materiales físicos para expanderse y contraerse sin llegar a romperse [Thevenin and Coutaz, 1999]. Al referirse a la plasticidad de las interfaces gráficas de usuario, estamos hablando de la capacidad de la intefaz para soportar variaciones por las características que presenta el sistema y el ambiente en el que se desenvuelve, pero siempre preservando la usabilidad. Así que un sistema que maneja plasticidad, siempre deberá presentarse usable al usuario, sin importar lo que él haga, el tipo de dispositivo, y siempre debe de estar consciente del contexto en el que se desenvuelve. Por ejemplo, el usuario de un sistema podría cometer un error de uso dentro del mismo, pero el sistema deberá responder de manera correcta a este, presentando una retroalimentación al usuario que lo guiará a tomar las mejores decisiones dentro del sistema, ya que el usuario no sabe si necesita forzosamente algo como acceso a red y cuando éste use la aplicación y el acceso a la red sea nulo, entonces la aplicación no funcione. El sistema debe estar consciente de su funcionamiento en los diferentes escenarios en los que puede encontrarse.

La plasticidad tiene dos formas de presentarse:

- Remodelación: que se refiere a los diferentes tipos de presentación que puede tener una interfaz de usuario, en base a variables de contexto como lo son, la actividad que este realizando la persona, como el dispositivo que está usando.

- Redistribución: referida a extender una interfaz de usuario entre diversos dispositivos que podrían tener diferentes plataformas (dispositivos heterogéneos).

En este trabajo de investigación se abordan ambos tipos de plasticidad, ya que el sistema colaborativo propuesto ajusta su interfaz dependiendo del dispositivo, aunque también puede compartir o extender la interfaz entre diversos usuarios. La aplicación es del tipo de edición de documentos no estructurados, ya que es una aplicación para generar diagramas o mapas mentales de manera colaborativa. Esta aplicación se decidió porque es un buen ejemplo de colaboración entre personas y puede ser usado por todo tipo de usuario, así que se intenta llegar al mayor número de usuarios. Cabe destacar que lo más importante en esta investigación, es la implementación del mecanismo de redistribución plástica, porque es una de las áreas menos exploradas dentro de los sistemas colaborativos.

1.2. Planteamiento del problema

Como se ha mencionado anteriormente, existen un gran número de dispositivos heterogéneos que cada vez son más diferentes, ya que siempre se crean nuevas plataformas que son más eficientes que sus predecesoras, logrando una mejora en el poder de cómputo, pero generando una diversificación que hace que los desarrolladores de software batallen para aprender e implementar nuevas tecnologías. Por lo tanto, las aplicaciones colaborativas deben de ser desarrolladas en varios ambientes de programación, específicos a cada plataforma. Las aplicaciones cuentan con una amplia gama de funcionalidades, que para el usuario comúnmente se ven reflejadas en componentes gráficos, como menús, iconos, u objetos específicos a la aplicación.

Así mismo, los usuarios deben poder usar sus aplicaciones desde cualquier tipo de dispositivo con el que ellos cuenten, por lo tanto la misma aplicación debe de ocultar o sustituir componentes gráficos dependiendo del dispositivo que se está utilizando. Algunas razones más para mostrar o sustituir los componentes son:

- Mostrar solo los iconos relacionados con la actividad que está haciendo el usuario, para que su atención se centre en lo que está haciendo solamente.

- Ocultar componentes cuando estos no tengan funcionalidad debido a que el recurso humano o físico no está disponible, e.g., el icono de una impresora debe ocultarse cuando está descompuesta o el icono de un colaborador no debe mostrarse cuando dicho colaborador no esté dentro del sistema;

Esto solo tiene que ver con la remodelación plástica de la interfaz de usuario, ya que solo se habla de como debe comportarse la interfaz de usuario, pero tomando en

cuenta que solo nos encontramos en una sola plataforma o dispositivo. El reto mayor es cuando contamos con diversos dispositivos y queremos que los usuarios puedan apoyarse de todos los dispositivos con los que cuentan para poder realizar una tarea de manera co-localizada, es decir, en el mismo lugar.

Este tipo de sistemas son los menos explorados, ya que se tienen que tomar en cuenta más factores sociales dentro de la implementación de este tipo de sistemas colaborativos. Por ejemplo, la distancia en la que los usuarios pueden trabajar, la red de comunicación que usan los dispositivos, la transición que existe cuando un usuario entra o sale del grupo. Podemos llamar a el espacio virutal en el que interactúan los usuarios mediante sus dispositivios, como espacio compartido.

Johanson, Fox y Winograd [Johanson et al., 2002] determinaron dos actividades básicas que los usuarios llevarían acabo en un sistema colaborativo que cuenta con espacio compartido:

1. *Mover datos*. Los usuarios requieren mover datos entre las pantallas de los diferentes dispositivos que tienen ejecutan la misma aplicación colaborativa.

2. *Libre movimiento*. Para minimizar interrupciones en una sesión colaborativa, cualquier usuario puede controlar cualquier dispositivo que esté ejecutando la aplicación.

Por lo tanto, esta investigación no tiene muchos trabajos que existan comercialmente o que un usuario común pueda obtener. Existen sistemas colaborativos con áreas compartidas, pero son pocos los trabajos que explotan el trabajo de manera co-localizada, ya que se tiene la costumbre de que siempre estamos conectados a la red más grande de todas, la Internet, por lo que muchas veces no se requiere de estar presentes en un mismo sitio para colaborar, pero se piensa que la interacción cara a cara es más productiva que cuando los usuarios no están en contacto físico.

1.3. Objetivos del proyecto

Objetivo general

Desarrollar un sistema colaborativo que pueda extender su espacio de trabajo, formando un espacio compartido entre dispositivos heterogéneos, y que adapte su interfaz de usuario para cada tipo de dispositivo, ya sea ocultando, sustituyendo o aumentando sus componentes gráficos.

Objetivos particulares

1. Diseñar y desarrollar la aplicación usando metodologías de software ágiles.

2. Extender la interfaz de usuario de una aplicación entre varios dispositivos (redistribución plástica).

3. Ocultar o sustituir componentes de la interfaz de usuario, dependiendo del dispositivo que se use (remodelación plástica).

4. La aplicación desarrollada deberá incorporar la redistribución y remodelación de manera efectiva.

1.4. Organización de la tesis

El trabajo de tesis esta estructurado en cinco capítulos. Se inicia dando una introducción a conceptos, para después pasar a la aportación que se ha hecho con el sistema colaborativo desarrollado.

El capítulo 2 contiene todos los conceptos necesarios para entender de que trata el trabajo de investigación. Se incluye todo sobre la plasticidad de las interfaces gráficas de usuario, se habla más a detalle de los tipos de plasticidad, así como también se aborda más a fondo la consciencia de contexto, porque es importante para la plasticidad. En el mismo capítulo se muestran los trabajos relacionados y más recientes, se habla de que fue lo que los autores aportaron y que relación tiene con el presente trabajo.

En los capítulos 3 y 4 se habla totalmente de la aportación que se hizo con el trabajo de investigación. En el capítulo 3, se mencionan las diversas tecnologías que se exploraron durante la investigación, para intentar llevar a cabo la implementación, ya que este trabajo fue iterativo y de mucha experimentación con diversas tecnologías, que podían apoyar en la comunicación, ó interacción de la aplicación. También se explica el diseño que se realizó para realizar el sistema, incluyendo algunos diagramas para un entendimiento más fácil de la abstracción del sistema.

En el capítulo 4 se habla totalmente de la implementación, desde la tecnología elegida hasta la descripción de algunos de los algoritmos que componen el sistema. Se explica la arquitectura elegida y los frameworks que fueron de ayuda en el desarrollo del sistema. También se muestran las pruebas que se efectuaron para la verificación y validación del sistema.

Por último, en el capítulo 5 se encuentran las conclusiones, con las cuales se discuten los problemas que se encontraron durante toda la investigación, así como las soluciones que se dieron, el porque se decidió hacerlo de cierta forma. Finalmente, se dejan abiertas algunas áreas de oportunidad en las cuales se puede seguir explorando e investigando. Se listan algunas ideas que pueden servir de mejora para el sistema colaborativo implementado, o para algún sistema parecido.

Capítulo 2

Fundamentos y Estado del Arte

El área de IHM (Interacción Hombre-Máquina) tiene una correlación directa con la ingeniería de software, ya que para llevar a cabo los desarrollos en ésta área, se necesita de tener una metodología que funcione para este tipo de investigaciones. Una de estas metodologías es conocida como Ingeniería Basada en Modelos, ya que trata de abstraer los conceptos que tenemos de las cosas en nuestro alrededor, para que podamos manipularlos dentro de un sistema de cómputo. En este capítulo...

2.1. Plasticidad

Debido a la proliferación en aumento de los dispositivos de cómputo, se ha encontrado la necesidad de que las aplicaciones cumplan con dos características importantes [Calvary et al., 2001]:

1. que sean independientes de la plataforma,

2. que se ejecuten en diferentes entornos físicos.

Generar una interfaz de usuario específica para cada contexto de uso es demasiado costoso. Así que es de suma importancia el usar el contexto de uso para la generación de la interfaz de usuario, y tomar en cuenta las característias antes mencionadas.

El término de plasticidad esta basado en las propiedades de los materiales para expanderse o contraerse bajo circunstancias naturales sin quebrantarse, preservando su uso constante. Dentro del área de IHM, puede definirse como *la capacidad de un sistema colaborativo para resistir las variaciones del contexto de uso, y así preservar su*

usabilidad [Thevenin and Coutaz, 1999].

En la ingeniería de software, se considera a la usabilidad como una propiedad intrinseca del producto de software, mientras que en IHM, la usabilidad no es intrinsicamente usable o no usable, más bien la usabilidad se presenta relativamente al contexto de uso [Coutaz and Calvary, 2012].

2.1.1. Adaptación

La adaptación, en relación a la IHM, se basa en dos propiedades del sistema: la adaptabilidad y la auto-adaptación. La adaptabilidad es la habilidad del sistema que permite a los usuarios personalizar su sistema a partir de un conjunto predefinido de parámetros. La auto-adaptación es la capacidad que tiene el sistema para adaptarse automáticamente sin ninguna acción por parte del usuario.

El espacio de diseño para la adaptación incluye tres ejes ortogonales más: objetivo, medios y tiempo [Thevenin and Coutaz, 1999].

El objetivo de la adaptación indica las entidades para las cuales se destina la adaptación: adaptación a los usuarios, adaptación al entorno y la adaptación a las características físicas del sistema.

Los medios de adaptación denotan los elementos del software del sistema que participa en la adaptación: el modelo del sistema de tareas, las técnicas de representación y los subsistemas de ayuda, pueden modificarse para adaptarse a las entidades seleccionadas.

En el eje de adaptación del tiempo, la adaptación puede ser estática, es decir, eficaz entre sesiones, y/ó dinámica, que ocurre en tiempo de ejecución.

2.1.2. Medios de adaptación

Remodelación

La remodelación se refiere al cambio de forma de la interfaz gráfica de usuario, aplicando transformaciones en algunas o todas las secciones de la interfaz gráfica. Las transformaciones incluyen [Coutaz and Calvary, 2012]:

- Suprimir componentes de la interfaz de usuario que se hacen irrelevantes a la

nueva situación o contexto de uso, incluyendo la actividad que podría estar llevando a cabo el usuario.

- Inserción de nuevos componentes a la interfaz de usuario para proveer acceso a nuevos servicios relevantes a la situación o contexto de uso.

- Substitución de compontentes de la interfaz de usuario cuando obviamente algunos componentes reemplazan a otros. Este paso puede ser visto como una combinación de los dos anteriores.

- Reorganización de los componentes de la interfaz de usuario. Esta transformación se puede llevar a cabo por los diversos tamaños con los que cuentan los dispositivos de cómputo por medio de los cuales los usuarios están interactuando con el sistema.

Redistribución

La redistribución de la interfaz gráfica de usuario denota la re-localización de los componentes de la interfaz de usuario hacia diferentes recursos de interacción, por ejemplo, hacia algún otro dispositivo. La granularidad de la redistribución de la interfaz de usuario puede variar de nivel de aplicación a nivel de pixel [Coutaz and Calvary, 2012]:

- En el nivel de aplicación, la interfaz de usuario es replicada totalmente en cada dispositivo de cómputo. Cuando la redistribución es dinámica, toda la interfaz de usuario migra a un nuevo dispositivo, por lo cual podría también ocurrir una remodelación.

- En el nivel de espacio de trabajo, la unidad que se distribuye es el mismo espacio de trabajo. Un espacio de trabajo es un espacio lógico o virtual, que puede ser visto por los usuarios como el lugar en donde llevan a cabo sus tareas.

- En el nivel de interactor, la redistribución es un caso especial del espacio de trabajo, ya que la unidad que se migra es solamente un componente o interactor principal.

- En el nivel de pixel, cualquier componente de la interfaz de usuario puede ser particionado entre diversos dispositivos.

2.2. Contexto de uso

El *contexto* ha sido usado en muchas áreas de la computación, creando campos como el cómputo consciente del contexto, y es necesario en los sistemas ubicuos, específicamente en sistemas colaborativos. Desde que el término se empezó a usuar, han existido muchas definiciones y una de las más aceptadas es la siguiente: *"El contexto es cualquier información que pueda ser usada para caracterizar la situación de una entidad. Una entidad es una persona, lugar, u objeto que sea considerado relevante en una interacción entre un usuario y una aplicación, incluyendo al mismo usuario y las aplicaciones"* [Dey, 2001].

Esta definición claramente menciona que el contexto siempre esta atado a las entidades y que la información que describe a la situación de una entidad es contexto. El uso de términos como información hace que la definición aún quede general. Por lo tanto, es necesario ser más explícitos en cuanto a lo que se trata de decir con información. Los elementos para describir a la información del contexto pueden entrar en cinco categorías: [Zimmermann et al., 2007] individualidad, actividad, localización, tiempo, y relaciones. La actividad determina de manera predominante la relevancia de los elementos de contexto en situaciones específicas, mientras que la localización y el tiempo principalmente manejan la creación de las relaciones entre entidades y permiten el intercambio de información entre éstas.

Así mismo, se puede definir a un sistema consciente de contexto, como aquel que usa al contexto para proveer información relevante y/o servicios al usuario, donde la relevancia depende de la tarea específica del usuario [Dey, 2001]. Aquí ahora también se hablan de las tareas que lleva a cabo el usuario, y podemos extender la definición a sistemas colaborativos, aunque también debemos tomar en cuenta que ya no se trata de un solo usuario, sino que pueden existir varios usuarios interactuando con las aplicaciones que existen en el sistema.

En éste trabajo de investigación, no se toman toma en cuenta la colaboración que puede existir entre las personas, ya que una persona puede trabajar individualmente como dentro de un grupo de personas, y esta transición puede ocurrir repetitivamente durante el lapso de uso de la aplicación. La colaboración ya es tratada más a fondo, gracias a áreas como el TCAC (Trabajo Colaborativo Asistido por Computadoras), pero aún representa muchos retos que deben de ser solucionados para soportarla efectivamente por los sistemas colaborativos [Haake et al., 2010].

Por ejemplo, imagina que estas haciendo uso de un editor de texto colaborativo, por medio del cual puedes editar un documento ya sea de manera individual como con otras personas. Puede ser que tú como usuario estes editando un documento en el cual has invitado a otras personas a colaborar, y en cualquier momento cualquier

persona participante puede salir y entrar al documento, sin pedir aprobación alguna. Por lo tanto, el sistema debe de brindar retroalimentación amigable a los usuarios de lo presencia o ausencia de los usuarios que estan trabajando en el documento. Existen muchas formas de implementar este tipo de sistemas (i.e. [Tandler et al., 2001]), pero poco a poco se esta llegando a contar con buenas prácticas para realizar este tipo de aplicaciones.

En los años recientes, se han tratado de llegar a propuestas más comprensivas para la interpretación del contexto. Estas propuestas incluyen aspectos no físicos como el interés del usuario, tareas, o la interacción y pueden ir lo más extenso posible como considerar cualquier tipo de entidad o cosa como parte del contexto de uso [Haake et al., 2010].

Dentro de los sistemas colaborativos, se deben considerar los siguientes componentes para un modelo conceptual del contexto [Haake et al., 2010]:

- Información sobre la situación o estado actual, que puede ser provisto por componentes que actuen como sensor.
- Información sobre el dominio de la aplicación.
- Reglas de contextualización para generar un estado de contexto.
- Reglas de adaptación que definan un conjunto de adaptaciones significativas de acuerdo al estado de contexto.

Dentro de los sistemas colaborativos plásticos se necesita tener un concepto más específico del contexto, ya que el contexto de uso puede llegar a perderse por tantos tipos de clasificaciones que se han hecho.

El contexto de uso en las interfaces gráficas de usuario plásticas se definen por tres entidades [Calvary et al., 2001]:

1. el *usuario* que conceptualiza a la persona que esta interactuando con el sistema,
2. la plataforma, que se refiere al software y hardware utilizados dentro de la interacción con el sistema,
3. el entorno que describe las condiciones físicas y sociales en donde se lleva a cabo la interacción.

2.3. Proceso de adaptación

La adaptación de la plasticidad puede ser estructurada por medio de cinco pasos [Calvary et al., 2001]:

1. Detección de las variaciones en el contexto de uso.

2. Identificación de las posibles interfaces de usuario adecuadas para el nuevo contexto de uso.

3. Selección de la interfaz de usuario.

4. Transición de la interfaz de usuario actual a la seleccionada como nueva solución.

5. Ejecución de la nueva interfaz de usuario hasta que las próximas condiciones de adaptación ocurran.

Los pasos anteriores pueden ser realizados por el sistema, el usuario, o ambos. En cualquiera de los casos se identifican tres tipos de cambios:

- El sistema tiene la capacidad de realizar los cinco pasos sin interverción del usuario, por lo que el sistema tiene la capacidad de hacer plasticidad *adaptativa*.

- El usuario realiza los cinco pasos manualmente, por lo tanto, el sistema hace uso de la plasticidad *adaptable*.

- Por último, la plasticidad *mixta* contempla la combinación de el usuario y el sistema.

2.4. Trabajos relacionados

Cada uno de los trabajos listados en esta sección es analizado para saber cómo hacen uso de la plasticidad, aunque hay algunos que simplemente usan técnicas que sirvieron de inspiración para el desarrollo de ésta tesis. Estos trabajos implementan una aplicación que funciona de manera individual, aunque algunas también cuentan con una parte colaborativa, permitiendo poder ser usada entre varias personas.

2.4.1. ConnecTables

ConnecTables [Tandler et al., 2001] es un sistema colaborativo en el cual los autores personalizaron y adaptaron dispositivos móviles, a los que llamaron ConnecTables, en los cuales ejecutan una aplicación para hacer dibujos. El uso de la aplicación entre dos personas, permite la redistribución plástica de la misma, haciendo que la interfaz de usuario se expanda entre los dos dispositivos. Esta interacción sólo es posible cuando los usuarios están en un mismo lugar, porque los dispositivos ConnecTables cuentan con sensores que permiten la expansión del espacio de trabajo.

Un dispositivo ConnecTable es una tablet equipada con un stylus para poder interactuar con la pantalla táctil y tiene incorporados sensores que permiten interpretar acciones físicas de lo que ocurre a su alrededor, haciendolo un dispositivo consciente del contexto. Por ejemplo, una de estas acciones físicas, es la aproximación de otro dispositivo ConnecTable con el cuál se puede interacutar.

Cuando dos dispositivos ConnecTable están cercanas entre sí, como se muestra en la figura 2.1, se crea una área de trabajo compartida temporal, combinando las áreas que antes eran personales a cada usuario. La área de trabajo temporal que se ha generado, permite a los usuarios trabajar entre los diversos dispositivos, brindando un espacio de trabajo más amplio ya que éste existe por la combinación de las pantallas de los dispositivos. Por lo tanto, se puede tener interacción usando los dos dispositivos, como se muestra en la figura 2.1a, en donde se pasan objetos manejados por la aplicación, como si de verdad los dos dispositivos formaran un mismo espacio de trabajo, o bien, fueran uno sólo. Esta área de trabajo compartida seguira funcionando hasta que los dispositivos ConnecTable se separen cierta distancia entre sí, con lo que cada dispositivo volverá a tener su propia área personal de trabajo.

(a) (b)

Figura 2.1: Los usuarios pasan objetos entre sí usando dos ConnecTables (figura 2.1a), después cada uno ve la figura que han intercambiado en su propio dispositivo (figura 2.1b).

La comunicación entre dispositivos ConnecTable se da por medio de una conexión inalámbrica. No se especifíca si se utilizó la tecnología *peer to peer* o cliente - servidor,

aunque la segunda opción es lo más posible, ya que si usaran *peer to peer* sería motivo de explicación.

Para saber si un dispositivo está cerca de otro, cada ConnecTable cuenta con una etiqueta con tecnología de radio frecuencia. Por medio de un alambre de cobre que rodea el dispositivo, se crea un campo magnetico. Cuando la etiqueta está dentro del radio de este campo, ésta manda una señal de vuelta, que es captada por el sensor, el cual procesa la señal. Este proceso de identificación se puede llevar a cabo cuando se está a una distancia aproximadamente de 3 cm entre ConnecTables.

Al hacer la redistribución de la aplicación, no se aprecia fácilmente si también se hace remodelación plástica. Al generar el área de trabajo compartida, uno de las aplicaciones cambia su perspectiva de uso y se gira 180 grados para que lógicamente, el sistema interprete que ambas aplicaciones están en la misma perspectiva, aunque físicamente parece no haber ningún cambio. Es imporante tener en cuenta este tipo de detalles, ya que pueden simplificar mucho el diseño y desarrollo de una aplicación de este tipo.

2.4.2. Pizarrón colaborativo

En el trabajo de [Morales, 2009] se creó un pizarrón colaborativo. Esta aplicación cuenta con un área de trabajo común, que es en donde todos los usuarios que estén dentro del sistema pueden dibujar. Este sistema hace uso la consciencia de grupo, por lo cual reconoce la presencia de los usuarios, ya que muestra si están conectados en una barra. La aplicación permite redistribuir su interfaz de usuario en los diferentes dispositivos que esté usando un mismo usuario, es decir suponiendo que un usuario cuenta con una PC y una PDA, éste puede iniciar una sesión usando ambos dispositivos, permitiendo redistribuir la interfaz de usuario, mostrando algunos componentes en la PDA y otros en la PC, como se muestra en la figura 2.2. Por ejemplo, en la PDA se puede mostrar la barra de herramientas que permite cambiar el color de una figura, mientras que en la PC se hace uso de esta opción dibujando en pantalla con el color que se ha escogido en la PDA.

Esta aplicación permite al usuario poder trabajar de manera individual o colaborativa según sus necesidades, pero la redistribución solo se presenta si cuenta con más de un dispositivo. Se ha tomado mucho en cuenta la redistribución que hace este trabajo, ya que sus técnicas son de ayuda para este trabajo de tesis.

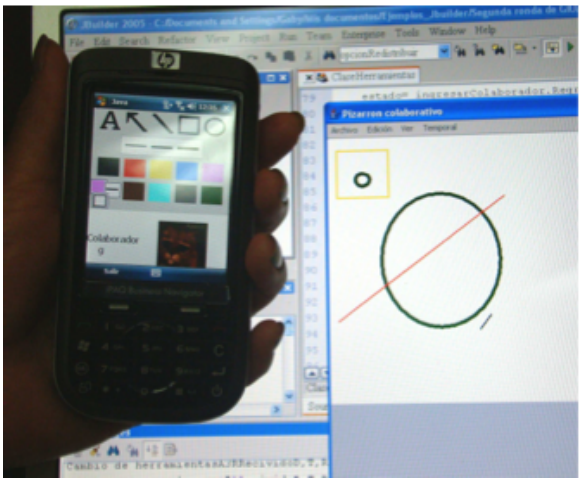

Figura 2.2: El dispositivo móvil muestra una barra de herramientas y barra de presencia, mientras que la PC muestra el área de dibujo.

2.4.3. UbiDraw

Ubidraw [Vanderdonckt and Calleros, 2008] es una aplicación de dibujo, que puede parecer simple porque solo es individual, por lo que no existe ningún tipo de colaboración entre múltiples usuarios. Lo importante de este trabajo es la remodelación plástica que lleva a cabo, porque usa diferentes tipos de algoritmos que toman en cuenta muchas consideraciones para acomodar la interfaz de usuario.

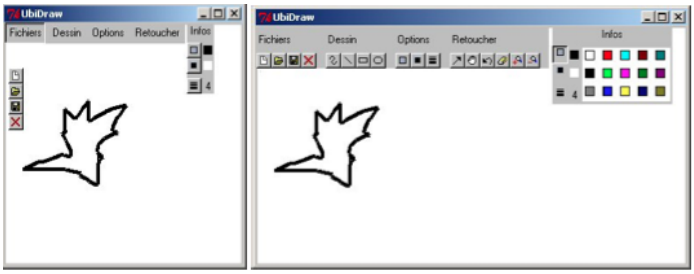

Figura 2.3: Diferentes tomas de pantalla de la aplicación, con diferentes tamaños de la ventana.

Para llevar a cabo la remodelación, el sistema toma en cuenta variables como el tamaño de pantalla, las preferencias del usuario, y la frecuencia de uso de ciertos componentes de la aplicación. En la figura 2.3 se puede apreciar como se muestran diferentes componentes en la interfaz de usuario, al contar con un tamaño de pantalla diferente.

Todo esto se toma en cuenta para que el usuario siempre tenga disponible un espacio de trabajo que sea práctico para él, porque la prioridad del sistema es que el usuario

pueda trabajar de manera eficiente, teniendo todos los componentes que él usa al alcance, pero sin estorbarle o que no se sienta con el espacio suficiente para poder trabajar.

Este tipo de implementación es única, porque aunque sistemas modernos como Android, ya permiten crear diferentes interfaces dependiendo del tamaño de pantalla, aún es responsabilidad del desarrollador decidir que componentes mostrar en pantalla, y tomando en cuenta algoritmos como los que usa UbiDraw, se pueden implementar sistemas que son muy amigables con el usuario.

Es importante recalcar que este trabajo sólo hace uso de la remodelación, así que lo único para lo que sirve al trabajo de tesis que se presenta en este documento, es para utilizar las técnicas que se describen en UbiDraw para mostrar la mejor interfaz de usuario en base a las características del dispositivo, así como también por lo que el usuario necesita.

2.4.4. Superficies aumentadas

En el trabajo de Superficies Aumentadas [Rekimoto and Saitoh, 1999] se extiende el espacio de trabajo de los diferentes dispositivos que estén en uso, por lo tanto emplea la remodelación, pero a un nivel que no es muy común. En este análisis se dejan fuera otras técnicas que los autores emplean, porque abarcan otras áreas, como visión por computadora.

Se hace uso de diferentes dispositivos y se cuenta con una mesa interactiva que actúa como el medio de interacción principal entre todos los demás dispositivos. En la figura 2.4 se puede apreciar como se pasa un objeto entre una laptop y la mesa interactiva. Los autores llamaron a este tipo de interacción *hyperdragging*, porque es como arrastrar el mouse de la computadora fuera de la pantalla llevando al objeto hacia otro dispositivo.

Esto hace uso de la redistribución plástica, porque genera un ambiente virtual compartido entre los diversos dispositivos. Esta técnica de interacción es muy importante, porque el *hyperdragging* es tomado en cuenta como un tipo de interacción para llevar a cabo de este trabajo de tesis.

2.4.5. MindMap

MindMap [Lucero et al., 2010] es una aplicación que permite a los usuarios trabajar colaborativamente para crear, editar, y vizualizar notas, simulando el uso que hacemos

Figura 2.4: Usando Hyperdragging para mover objetos entre dispositivos.

de los Post-it o pequeños cuadros de papel para recordarnos sobre cosas que debemos hacer.

Este sistema usa la remodelación de su interfaz de usuario, porque permite extender el espacio de trabajo entre varios dispositivos. Para compartir un espacio de trabajo entre diferentes dispositivos, se hace uso de gestos que son interpretados por el sistema. El gesto que se usa es el de agarre, como se muestra en la figura 2.5. Con este gesto, podemos saber la posición en la que se encuentra un dispositivo con respecto a otro, y así armar un espacio de trabajo en base a las necesidades del usuario.

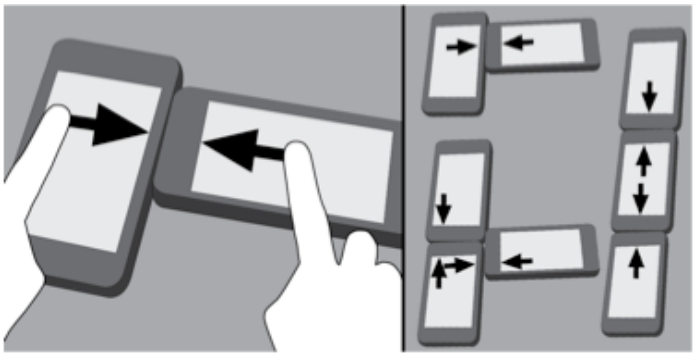

Figura 2.5: Gesto de agarre para juntar dispositivos y diferentes posiciones en las que se pueden posicionar los dispositivos.

Para llevar a cabo este funcionamiento, se comunican todos los dispositivos por medio de una red ad-hoc Wi-Fi, así como también uso de Bluetooth para aquellos dispositivos que no se encuentren conectados a la red Wi-Fi, lo que genera una red híbrida entre Wi-Fi y Bluetooth, para permitir la operación entre cualquiera de estos medios de comunicación.

Este mecanismo de remodelación solo puede ser usado con dispositivos con pantallas táctiles, ya que se necesita de gestos para saber la posición en la que se encuentra un dispositivo en relación a otro. Se puede experimentar con otro tipo de técnicas para dispositivos que no cuentan con pantalla táctil, como por ejemplo usando el acelerometro [Hinckley, 2003], que permitiría conocer también la posición en la que se encuentra un dispositivo vecino.

Capítulo 3

Análisis y diseño

En éste capítulo se hace un análisis y diseño a detalle de las partes que componen al sistema propuesto, el cual tendrá la función de desarrollar mapas mentales de manera individual como colaborativa. Se inicia describiendo la aplicación, dando un ejemplo de su uso en un caso real (sección 3.1). En la sección 3.2 se muestra la arquitectura para la aplicación en todos sus niveles, es decir, la arquitectura en el modo individual y colaborativo para la aplicación. Se pasa a explicar a detalle como funcionan los módulos más importantes de la aplicación en modo individual (seccion 3.3). Por último, la sección 3.4 describe la comunicación y algoritmos que existen para lograr la colaboración entre varios usuarios usando dispositivos heterogéneos.

3.1. Aplicación propuesta

Esta sección describirá qué es la aplicación que se realizó en el trabajo de tesis y dará un ejemplo de uso de la misma. También se explica en que tipo de sistema colaborativo se cataloga la aplicación.

3.1.1. Descripción de aplicación propuesta

La aplicación que se ha desarrollado en este trabajo de tesis, tiene como finalidad facilitar la transición del trabajo individual al trabajo colaborativo co-localizado. La co-localización se refiere al tipo de trabajo cuando dos o más personas se encuentran en el mismo lugar. Esta tesis tiene como interés el uso de las aplicaciones bajo este esquema de trabajo, ya que existen otros modos. La Figura 3.1 muestra los modos en

los cuales se puede estar trabajando colaborativamente [Ellis et al., 1991].

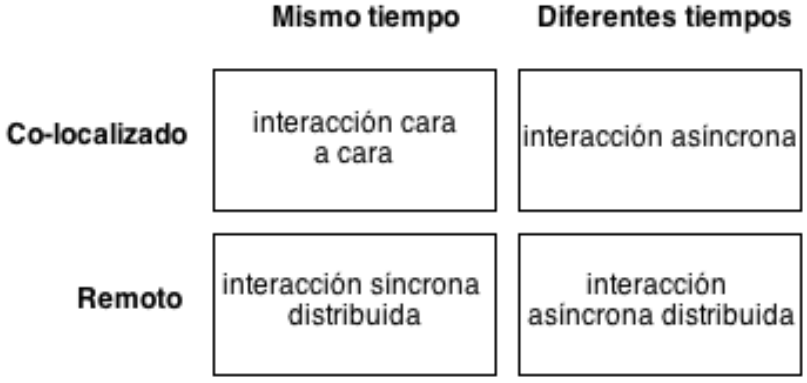

Figura 3.1: Matriz espacio-tiempo para sistemas colaborativos.

Puede haber tipo de colaboración de manera remota o co-localizada, así como también puede darse al mismo tiempo o en diferentes momentos. Por ejemplo, las aplicaciones que entran dentro de colaboración remota y al mismo tiempo, se encuentran todos los chats que conocemos, ya que nos estamos comunicando con una persona que no esta presente con nosotros pero interactuamos al mismo tiempo. Así mismo, algunas aplicaciones que entran en la clasificación de la colaboración remota en diferentes momentos, son editores de textos en línea, los cuales pueden ser editados por diferentes usuarios sin la necesidad de que todos los colaboradores estén trabajando en ese momento, cada usuario puede conocer los cambios en el documento que han hecho sus colaboradores en el momento en el que éste entre al sistema.

Por otra parte, también existen sistemas co-localizados, donde los colaboradores están en el mismo lugar. Este tipo de aplicaciones han sido menos estudiadas, y es por eso que se ha hecho este trabajo, ya que existen muchas áreas de oportunidad a ser explotadas en este campo.

El último caso, es decir, la colaboración co-localizada en diferentes momentos es un caso especial, ya que se lleva a cabo por medio de una interacción asíncrona. Por ejemplo, imagine que existe un operador monitoreando el estado de los semáforos en alguna ciudad. Él estará haciendo su trabajo sólo por el turno que debe, por lo tanto, después del turno de ésta persona, llegará alguien más a sustituirlo. Un sistema co-localizado que permita interacción asíncrona, facilitaría el trabajo de estas personas, ya que el usuario que termina su turno, puede dejar mensajes de lo que ha pasado, en este caso puede anunciar los semáforos que no funcionan, y así la persona que tome su lugar podrá saber el estado del sistema, y tomar las decisiones pertinentes.

Esta tesis se centra en el estudio de la colaboración co-localizada al mismo tiempo. La aplicación colaborativa propuesta es una herramienta para la generación de mapas mentales. Tiene la facilidad de ser usada de manera individual como colaborativa-

mente.

Con esta aplicación se pueden manejar varios proyectos de mapas mentales, y dentro de cada proyecto, se pueden agregar, modificar o borrar los nodos o elementos que componen un mapa mental. Está desarrollado para que el usuario haga uso de la aplicación de una manera natural. Existen herramientas que permiten la elaboración de mapas mentales o gráficos, y poder compartir el mapa mental con otras personas, sin embargo, no existe una herramienta que también permita la elaboración del mapa mental de manera grupal, en caso de que hayan más personas en el mismo lugar.

Para este proyecto se eligió la creación y manipulación de mapas mentales de una manera interactiva y natural, porque un diagrama en forma de mapa mental es usado por cualquier persona, desde estudiantes hasta profesionistas, como también para el uso común para cualquier persona que quiera plasmar sus pensamientos o ideas y poder acceder a ellas por medio de los dispositivos con los que cargamos a cualquier lugar. Es una manera divertida y eficiente de manifestar las ideas de las personas.

3.1.2. Escenario

El siguiente escenario pretende ejemplificar el uso del sistema colaborativo propuesto. Este sistema es para que los usuarios puedan crear y colaborar en una lluvia de ideas, haciendo mapas mentales y gráficos sencillos.

Supongamos que en una empresa le piden a un equipo conformado por tres personas, que desarrollen algunas ideas para un nuevo proyecto. Así que ellos deciden usar mapas mentales para la generación de ideas y relacionarlas. Para ello, están equipados con tablets y smartphones con capacidades de comunicación inalámbrica, y tienen instalada la aplicación propuesta en este trabajo de tesis para desarrollar mapas mentales colaborativamente.

Los miembros del equipo son Paulina, Adrian, y Miguel, cada uno tiene un dispositivo móvil con la aplicación instalada. Todos ellos se encuentran en el mismo edificio, y Paulina es la líder del proyecto. Entonces Paulina empieza a trabajar mientras que sus compañeros están haciendo algo más.

Por lo tanto, ella ejecuta la aplicación de mapas mentales y crea un nuevo proyecto para generar un mapa mental nuevo, en el cual trabajarán la idea que les ha sido pedida. Al crear el proyecto, se abre un área de trabajo que es donde el usuario, en este caso Paulina, debe poner los elementos que componen el mapa mental. Empieza a trabajar, agregando los nodos que conforman el diagrama, edita el texto de cada nodo y también tiene la posibilidad de agregar imágenes dentro de los nodos.

Después de un rato Adrian está listo para unirse al trabajo que Paulina ya ha empezado. Para comenzar a trabajar juntos, Adrian y Paulina ponen sus dispositivos uno al lado del otro, y entonces hacen un gesto de unión, tocando la pantalla y arrastrando su dedo desde su dispositivo hacia otro. Este gesto indica que desean trabajar juntos y entonces lo que ya había hecho Paulina hasta ese momento es compartido con Adrian. Cuando esto sucede, el área de trabajo en el que estaba trabajando Paulina, se redistribuye al dispositivo de Adrian, haciendo que el área de trabajo ahora sea más grande porque toma el tamaño de dos pantallas, con las cuales ambos colaboradores están trabajando. Desde este momento ambos pueden seguir interactuando mediante los dos dispositivos para la generación del mapa mental.

Minutos más tarde, Miguel también desea incorporarse al trabajo que han desarrollado sus compañeros. Entonces también pone lado a lado su dispositivo con el de alguno de sus compañeros. Él y Adrían hacen el mismo gesto de unión, lo cual indica al sistema que desean trabajar juntos. Por lo tanto, se redistribuye la interfaz de usuario entre los tres dispositivos, teniendo un área de trabajo aún mayor y permitiendo más visibildad del mapa mental. Los tres pueden estar interactuando a la vez, agregando más nodos, modificandolos y borrando lo que consideren pertinente.

Ahora Adrian desea moverse de lugar, para tener otra perspectiva del mapa mental. Para lograrlo, mueve su dispositivo a otro lugar, organizando de otra manera la forma en la que estaban posicionados tanto los dispositivos como las personas que están trabajando. Para cambiarse de lugar, Adrian debe indicar con un gesto de unión que se encuentra en la nueva posición, así que lleva a cabo el gesto con el dispositivo más cercano que tiene. Después de esta acción, el sistema vuelve a redistribuir la interfaz para que el área de trabajo del mapa mental ahora muestre de otra manera lo que se está viendo en ese momento, según la posición de los usuarios y sus dispositivos.

Más adelante, Paulina desea retirarse de esta sesión de trabajo, por lo que opta por indicar al sistema que desea terminar de colaborar con sus compañeros. Paulina puede terminar la sesión colaborativa mediante un componente en forma de botón que está presenten en la interfaz de usuario, el cual tiene la función de cerrar la sesión colaborativa y volver a una sesión individual. Ahora Adrian y Miguel siguen trabajando con los dispositivos restantes.

Llega el momento en el que están satisfechos con el trabajo que han hecho hasta ahora, por lo que ambos desean terminar el trabajo colaborativo que se ha realizado hasta ese momento. Ellos pueden optar por usar la mismo acción que ejecutó Paulina cuando se fue, o también podrían simplemente cerrar la aplicación y retirarse.

Cada miembro del equipo contará con una copia de este trabajo, ya que se guardará en un servidor el cual sabe que los colaboradores estuvieron trabajando juntos. Así que, aunque estén trabajando de manera individual, cada uno cuenta con lo último que

hicieron de manera conjunta.

3.2. Arquitectura de la aplicación

En la actualidad, uno de los patrones arquitecturales más usados para el desarrollo de sistemas es el patrón Modelo-Vista-Controlador (MVC). Este patrón nació junto a los primeros lenguajes de programación orientados a objetos, para proveer a los programadores una manera sencilla de separar la interfaz de usuario de la lógica de negocio del sistema [W. Greg Phillips, 1999]. Por lo tanto, es fácil utilizar MVC para abstraer las ideas del mundo real y transformalas en lenguaje que entienda la computadora.

La Figura 3.2 muestra un diagrama general de MVC, en el que cada nodo es un componente de MVC. El usuario de la aplicación interactúa con el componente de vista, el cual manda acciones al controlador, que está encargado de decidir qué hacer con base en lo que el usuario realizó. Podría manipular el modelo, el cual contiene toda la lógica y el estado del sistema y cuando existe un cambio, el modelo notifica al controlador del mismo. Por consiguiente, el controlador es el encargado de decirle a la vista que existen cambios y que debe actualizar la interfaz gráfica de usuario, la cual reflejará total o parcialmente el estado actual del modelo.

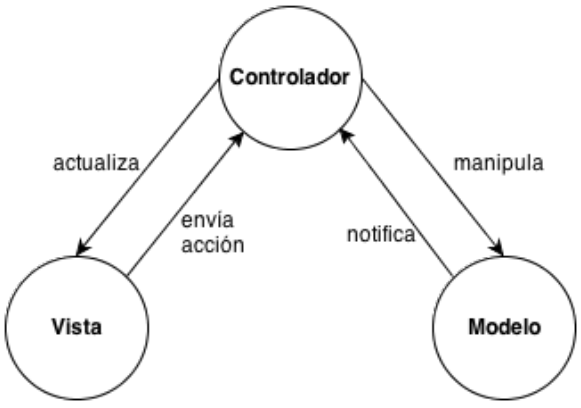

Figura 3.2: Vista general de MVC.

En el sistema de mapas mentales para el manejo de la interfaz de usuario, se unirán la *vista* con el *controlador*, mientras que el *modelo* será el encargado de controlar el estado y la lógica de todo el sistema.

3.2.1. Arquitectura individual

La Figura 3.3 representa el diagrama de clases conceptual de la aplicación cuando está en su modo individual. Se muestran las diferentes clases de la aplicación y cómo se relacionan entre sí.

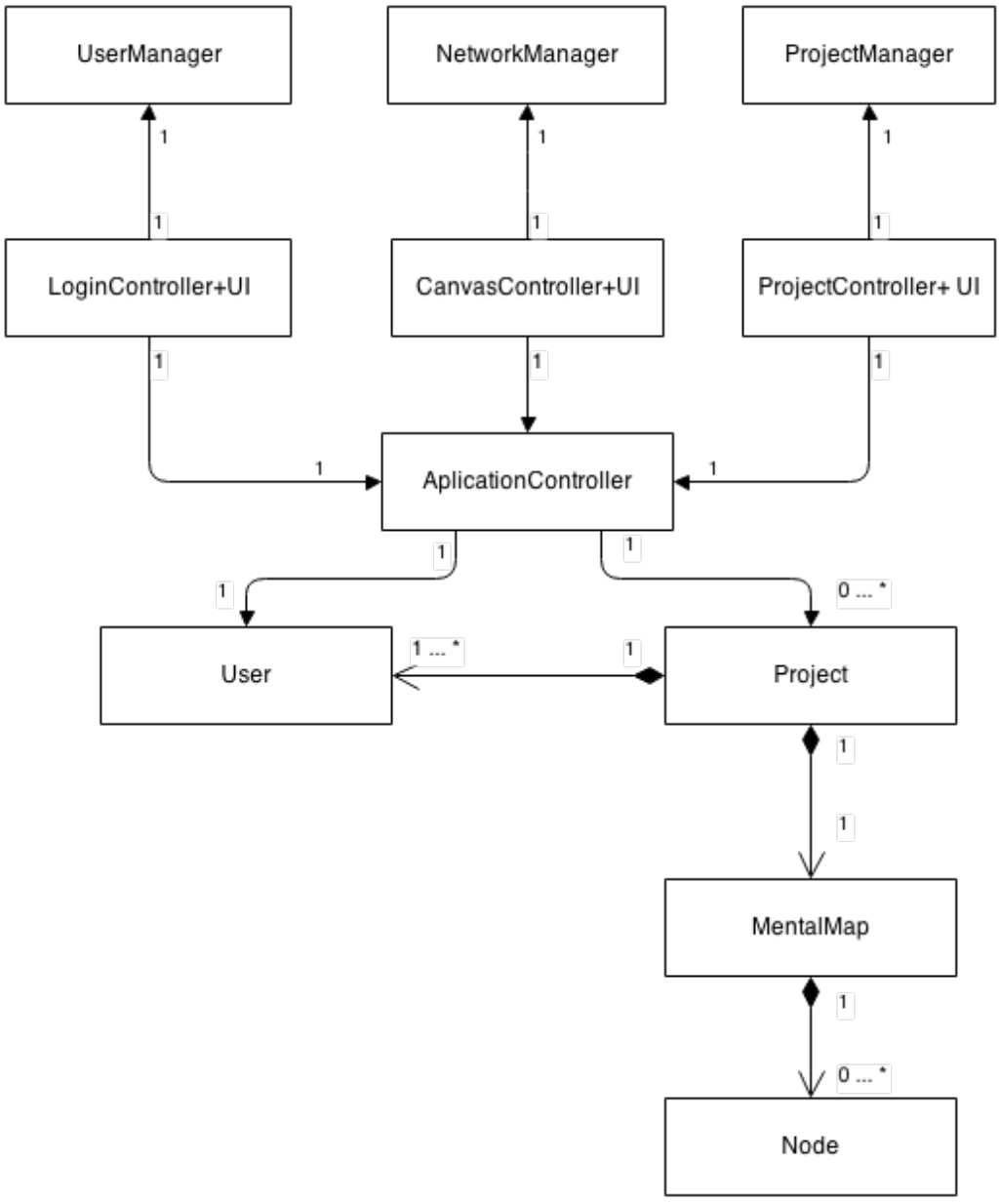

Figura 3.3: Diagrama de clases del modo individual.

A continuación se describen los componentes que aparecen en la Figura 3.3. Las clases con sufijo *Controller+UI* se refieren a los elementos que unen la capa de la vista y

controlador:

- **ApplicationController**: Es el controlador principal de la aplicación y mantiene el estado de ésta. Contiene el usuario (**User**) y los proyectos (**Project**) que se están usando en la aplicación.

- **LoginController+UI**: Maneja la interfaz de usuario para el inicio de sesión en el sistema. Cuando un usuario inicia sesión en el sistema, entonces este componente lo comunica a ApplicationController para que mantenga el usuario dentro de la aplicación.

- **ProjectController+UI**: Muestra el listado de proyectos que el usuario que inicio sesión previamente tiene permiso de usar.

- **CanvasController+UI**: Contiene el área de trabajo en donde los usuarios crean su mapa mental.

- **User**: Una clase que representa al usuario que usa la aplicación.

- **Project**: Representa el proyecto en el que trabajan usuarios sobre un mapa mental.

- **Mental Map**: Clase que maneja el estado de un mapa mental, incluyendo todos sus nodos.

- **Node**: Representa un nodo dentro del mapa mental, que expresa el contenido del mapa.

- **UserManager**: Se encarga de administrar el acceso de los usuarios a la aplicación, así como la creación de nuevos usuarios.

- **ProjectManager**: Administra la creación y eliminación de los proyectos para cada usuario de la aplicación.

- **NetworkManager**: Este componente es crucial para que la aplicación funcione de manera colaborativa, porque contiene los mecanismos para notificar a otros dispositivos los cambios que se hacen al mapa mental, escuchar cambios que se hacen al mapa mental en otros dispositivos, e iniciar o terminar una sesión colaborativa.

La aplicación esta dividida en tres módulos principales:

1. **Manejo de usuarios**: Es el módulo que se encarga de manipular todo lo re-

lacionado a los usuarios. Las clases principales que interactúan en éste módulo son *LoginController+UI*, *User*, y *UserManager*.

2. **Administración de proyectos**: Es el módulo encargado de crear, eliminar y mostrar los proyectos para un usuario. Las clases que apoyan al módulo son *ProjectController+UI*, *Project*, y *ProjectManager*.

3. **Área de trabajo y herramientas de dibujo**: Es el módulo mediante el cual los usuarios pueden manipular un mapa mental. Las principales clases que interactúan son *CanvasController+UI*, *Mental Map*, y *NetworkManager*.

En la sección 3.3 del capítulo, se explica mediante diagramas de secuencia como se mandan mensajes entre sí las clases antes descritas.

3.2.2. Arquitectura colaborativa

La aplicación entra en modo colaborativo cuando se está trabajando un proyecto usando dos o más dispositivos. En éste modo, uno de los dispositivos que están interactuando toma el rol de coordinador, que funciona como el servidor porque recibe todas las peticiones de los clientes dentro de la sesión colaborativa.

El coordinador no tiene interfaz de usuario, sólo contiene los mecanismos de comunicación y algoritmos para mantener un proyecto de forma colaborativa. La Figura 3.4 muestra las clases que se usan en el coordinador.

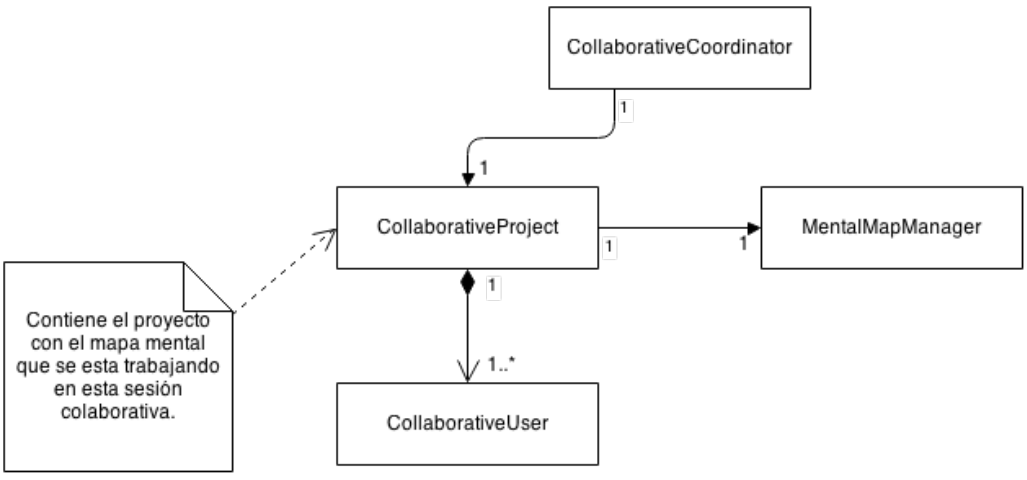

Figura 3.4: Diagrama de clases del modo colaborativo.

Ahora se describen cada componente de la Figura 3.4:

- **CollaborativeCoordinator**: Es el punto de entrada del coordinador con todos los clientes que interactúan en una sesión colaborativa. Recibe todas las peticiones de los clientes, y envía los cambios del mapa mental a cada cliente.

- **CollaborativeProject**: Mantiene el estado del proyecto que se esta compartiendo. El proyecto que maneja la clase está sincronizandose con lo que hace cada cliente en su dispositivo, por lo que cada cliente tiene la misma copia de este proyecto, lo que permite que cada uno pueda visualizar el proyecto en sincronía.

- **CollaborativeUser**: Contiene información de un usuario en la sesión colaborativa, como el nombre del usuario, el dispositivo que está usando y la resolución del mismo.

- **MentalMapManager**: Cuenta con todos los mecanismos para llevar a cabo la remodelación de la interfaz de usuario. A cada usuario se le asigna que parte del mapa mental puede visualizar, así que esta clase se encarga de asignar el espacio que visualiza cada cliente usando la resolución con la que cuenta el dispositivo y en donde se encuentra físicamente con respecto a los demás dispositivos.

En la sección 3.4 se explica a más detalle como funcionan los algoritmos que permiten trabajar de manera colaborativa, aquí sólo se introducen las clases del coordinador para que se tenga un vistazo de la arquitectura y ver como está organizada la aplicación.

3.2.3. Arquitectura global

En la Figura 3.5 se muestra la comunicación que existe entre el cliente y el coordinador. Esta figura contiene el diagrama de clases general de la aplicación.

En una sesión colaborativa, el *NetworkManager* dentro de un cliente envía cambios de su mapa mental al *CollaborativeCoordinator* que es parte del coordinador. Así mismo, el *CollaborativeCoordinator* siempre esta esperando peticiones de un cliente y cuando una llega, actualiza el estado del proyecto comunicando los cambios al *CollaborativeProject*. Una vez actualizado el mapa mental colaborativo, se envía el mapa sincronizado a todos los *NetworkManager* de los demás clientes que participan en la sesión colaborativa.

Aún falta mencionar que existe un componente más dentro de la aplicación, pero que sólo lleva a cabo el almacenamiento de toda la información de manera remota. Éste componente es un servidor con el cual los clientes se pueden comunicar cuando tienen

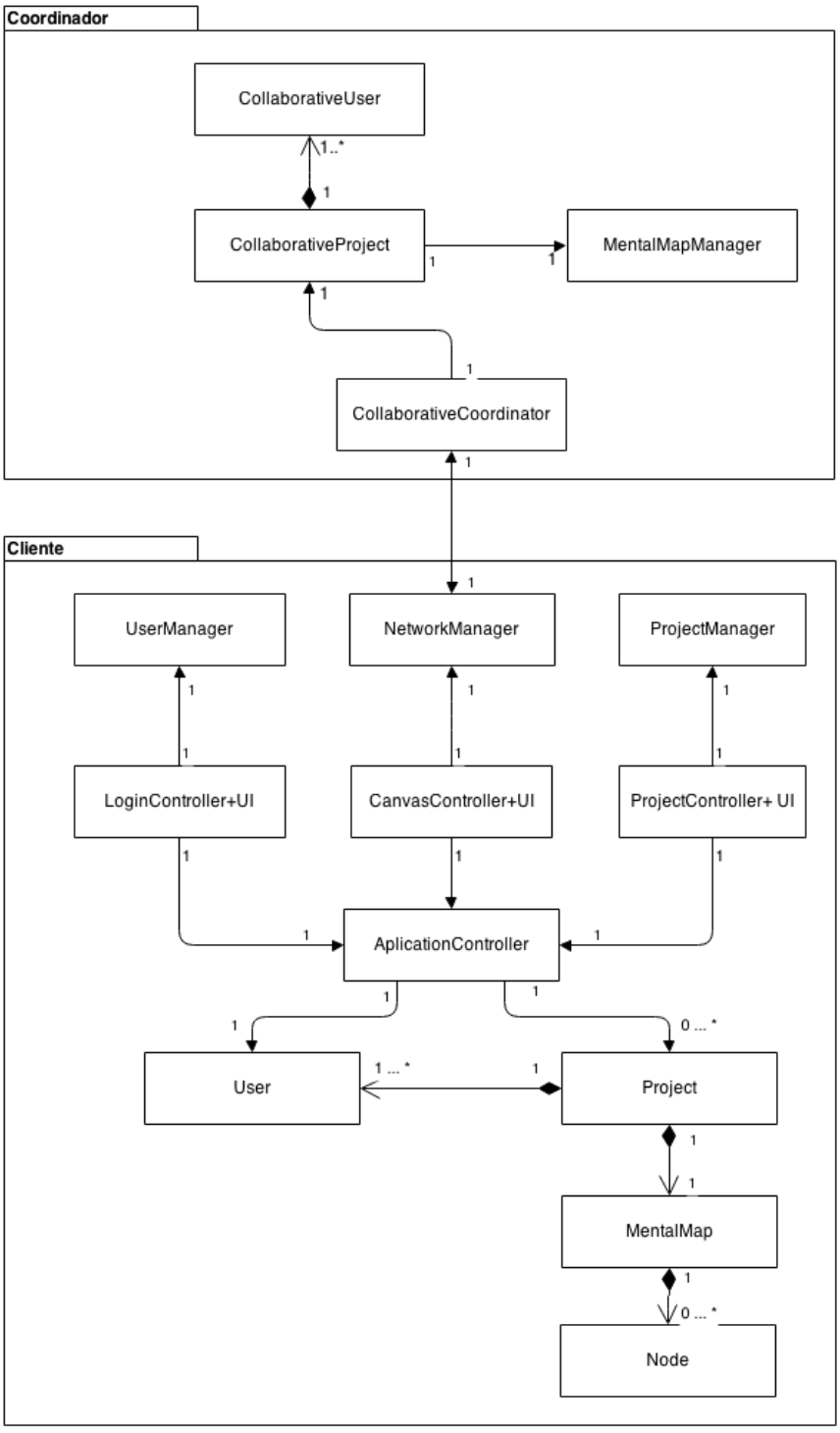

Figura 3.5: Diagrama de clases general del sistema.

acceso a Internet.

Su tarea es tener la información de todos los usuarios y proyectos que existen en la aplicación. Aquí se crean los usuarios y se guardan todos los proyectos que se comparten en la aplicación.

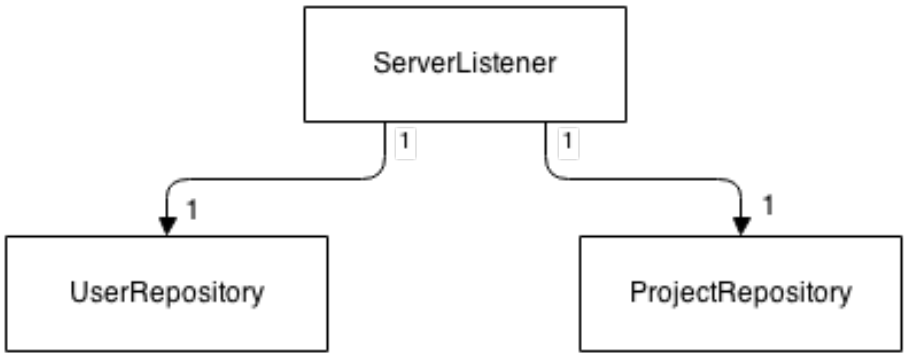

Figura 3.6: Diagrama de clases del servidor remoto.

La Figura 3.6 representa el diagrama de clases para el servidor remoto. A continuación se describen sus componentes:

- **ServerListener**: Escucha peticiones de un cliente. Ésta puede ser relacionada a un usuario o un proyecto.
- **UserRepository**: Se encarga de administrar todos los usuarios que existen en la aplicación.
- **ProjectRepository**: Se encarga de administrar los proyectos que se comparten en la aplicación.

Por lo tanto, cuando un cliente desea compartir alguno de sus proyectos, se envía el proyecto al servidor, pero sólo para guardarlo en un repositorio remoto y que los usuarios puedan acceder a éste con los cambios más recientes que existen en el proyecto. El servidor remoto no lleva a cabo ninguna operación para llevar a cabo el modo colaborativo del cuál se habló previamente, únicamente es un repositorio central de información.

3.2.4. Arquitectura de comunicación

El sistema de mapas mentales puede funcionar sin importar el estado de la conexión del dispositivo a Internet, ya que sólo se requiere estar conectado a una red local (LAN) en la cual estén todos los dispositivos que operarán de modo colaborativo.

Existen varias arquitecturas o topologías de comunicación que pueden funcionar para un sistema colaborativo como el desarrollado en este trabajo de tesis. Entre las más funcionales están *cliente-servidor* y *peer-to-peer*, ambas tienen ventajas y desventajas.

En cliente-servidor, cada dispositivo sólo hace cierta funcionalidad del sistema, ya que en el servidor es donde se lleva la lógica más robusta, como el control de concurrencia y la administración de las conexiones activas.Gracias al servidor, los clientes usan menos recursos de cómputo, que en dispositivos móviles es muy importante ser muy eficiente con estos recursos por la bateria limitada de los dispositivos.. La desventaja es la dependencia al servidor, ya que siempre debe existir un medio de conexión entre el cliente y el servidor, por lo que si esta conexión es interrumpida de alguna manera, el sistema dejará de funcionar en cierta forma.

Por otro lado, con peer-to-peer se remueve la dependencia a un servidor, ya que todos o algunos de los clientes se convierten en servidor. Lo anterior, da la ventaja en que no debe existir una conexión abierta a un servidor, pero la desventaja es que se pierde en el poder de cómputo necesario para llevar a cabo este tipo de arquitectura, y empeora cuando se implementa en dispositivos móviles, porque como lo mencionamos anteriormente, tienen recursos limitados.

En el sistema de mapas mentales, se optó en usar cliente-servidor. La Figura 3.7 muestra la arquitectura de comunicació que tiene el sistema. En la figura se da un ejemplo de colaboración entre tres dispositivos, y el que se encuentran en medio es el coordinador. El coordinador es el servidor cuando se establece una sesión colaborativa entre los tres dispositivos. Cada dispositivo cliente se comunica con el coordinador para sincronizar el mapa mental con el que estan trabajando.

Así mismo, cuando un dispositivo tiene acceso a Internet se puede comunicar con el servidor remoto para guardar u obtener información. Por ejemplo, cuando un usuario intenta registrar una nueva cuenta en la aplicación, es necesario que su dispositivo tenga acceso a Internet, ya que en el servidor remoto se tienen todas las cuentas de usuario para verificar que no existan cuentas con el mismo nombre de usuario.

También se puede apreciar que cada elemento tiene una base de datos, y esto es requerido porque cada uno tendrá su copia de información. Cuando un usuario usa la aplicación en su dispositivo, puede solicitar al sistema que guarde la información de su usuario de manera local, lo que permite tener acceso a su información de manera más rápida, ya que no tiene que comunicarse con el servidor remoto. Aún así, en el servidor existirán los datos de todo el sistema, mientras que en cada cliente sólo se tendrán los datos de interés para ciertos usuarios.

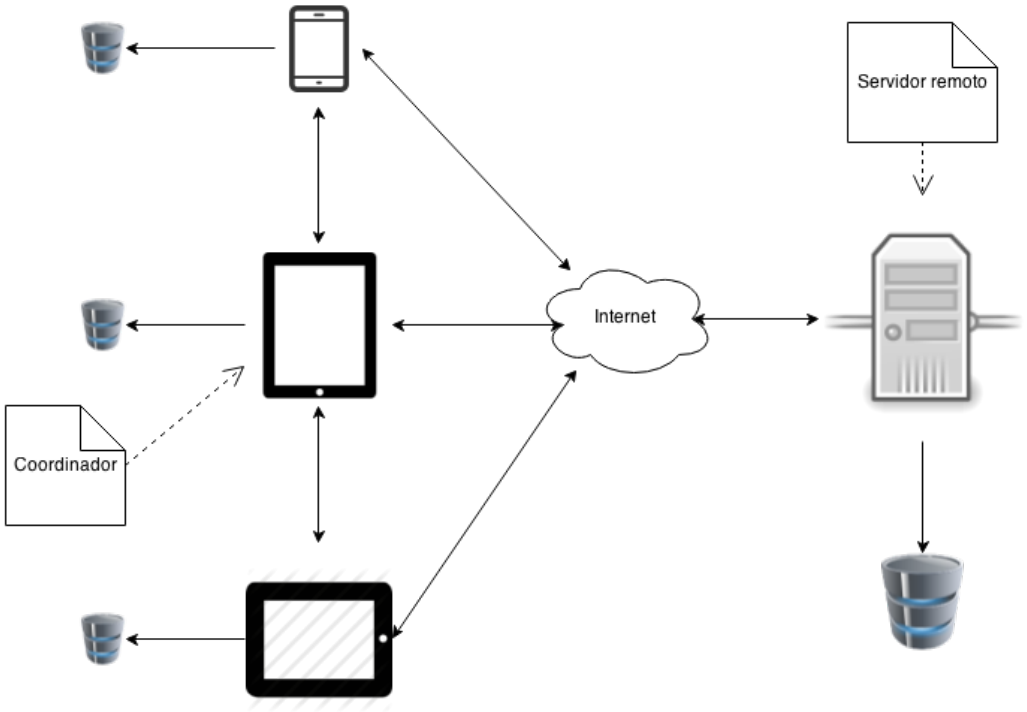

Figura 3.7: Diagrama de comunicación del sistema.

3.3. Modo individual

La aplicación inicia en su modo individual, ya que apenas se inicio el sistema y no se conoce nada de lo demás dispositivos. Por lo tanto, el modo individual cuenta con la funcionalidad necesaria para poder generar mapas mentales por una sola persona en su dispositivo, aunque no exista una conexión a una red.

3.3.1. Manejo de usuarios

Este módulo es necesario porque tiene la obligación de crear una cuenta para un nuevo usuario en el sistema, iniciar sesión de un usuario existente al sistema, y facilitar la función de salir del sistema cuando el usuario así lo requiera. Las tareas de este módulo se muestran en la Figura 3.8.

Podría extenderse la funcionalidad del módulo, por ejemplo, si se agregará la acción de poder eliminar un usuario existente en el sistema, pero el trabajo de tesis no satisface este requisito, así que no se implementó el mecanismo para borrar un usuario.

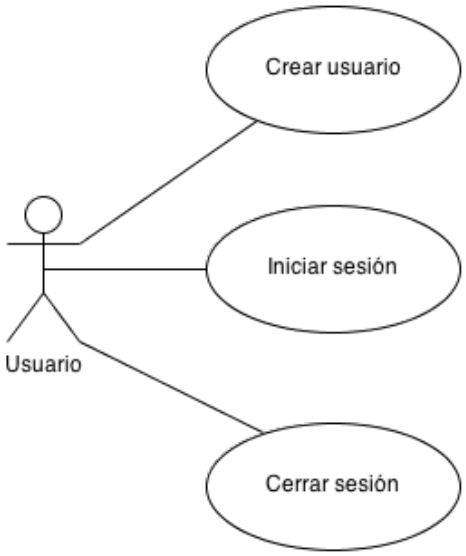

Figura 3.8: Caso de uso del manejo de usuarios.

Debe de existir una base de datos para persistir los datos, la cual guardará la información de cada uno de los usuarios en el sistema. En la Figura 3.9 se muestran los campos que necesita un usuario en la base de datos. Sólo se requieren dos datos por parte del usuario, los cuales son el nombre que se le dará a su cuenta y una contraseña. En la base de datos se guarda un registro por cada uno de los usuarios, cada registro contiene el nombre del usuario y la contraseña. El nombre del usuario es un campo único, por lo que no se podrá tener un nombre de usuario duplicado.

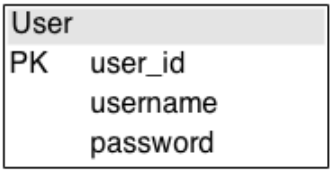

Figura 3.9: Tabla de base de datos de usuarios.

Cuando se inicie el sistema y aún no se haya iniciado sesión, se presentará al usuario una interfaz de usuario que preguntará por su nombre de usuario y contraseña, o también se le brindará la opción de poder dar de alta un nuevo usuario.

La Figura 3.10 muestra el diagrama de secuencia para la creación de un nuevo usuario. Este es el caso en el que la creación es exitosa, ya que también puede pasar que el usuario que se intenta crear ya existe, es decir, el nombre de usuario está duplicado, por lo que el sistema avisaría al usuario de que el usuario que intenta crear ya existe.

En el diagrama de la Figura 3.10 se puede ver como se comunica el cliente con el servidor remoto, cuando *UserManager* envía los datos del nuevo usuario a *Server-*

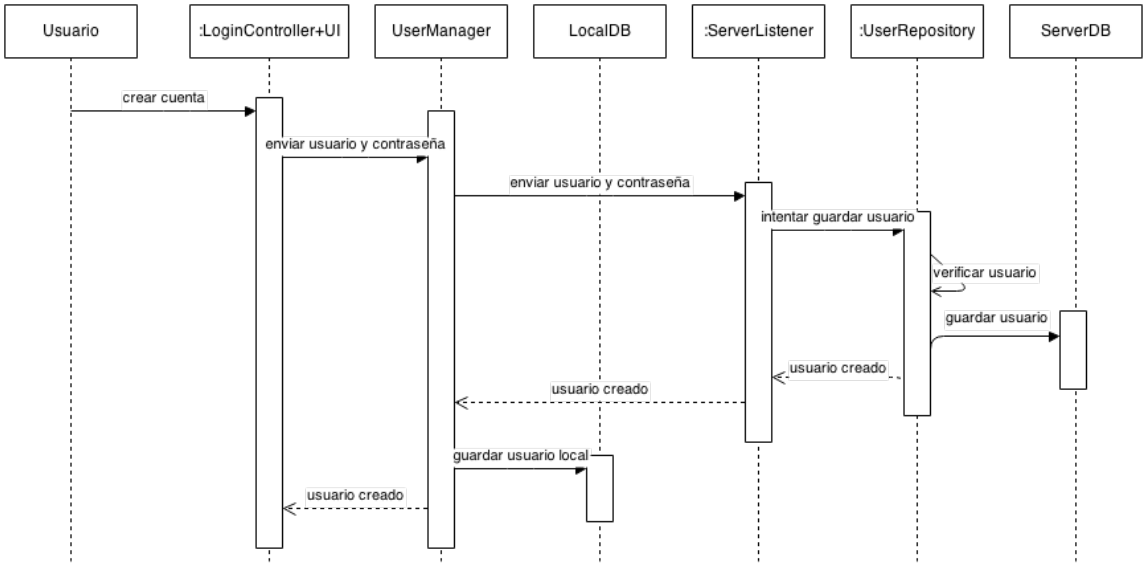

Figura 3.10: Diagrama de secuencia para crear un usuario.

Listener. En el servidor remoto se verifica la existencia del usuario, y si el nombre de usuario aún no existe, se guarda en la base de datos del servidor. Se regresa el usuario creado al cliente y en *UserManager* se puede guardar el usuario, que regreso el servidor, en la base de datos local.

En la Figura 3.11 se muestra el diagrama de secuencia para el inicio de sesión en el sistema. El usuario procede a ingresar sus datos en el sistema, y entonces *LoginController* se encarga de pasar los datos de inicio de sesión a *UserManager*, él cual accede a la base de datos para verificar que el usuario existe. En el caso de que no existiera, se devolvería al usuario la respuesta de que el usuario no existe, de lo contrario se procedería de manera normal. Cuando el registro del usuario se encuentra en la base de datos, se guarda la referencia a este usuario en *ApplicationController* para que poder trabajar dentro de la aplicación haciendo uso de la información de este usuario.

Para el cierre o finalización de la sesión, solamente se presentará un componente como un botón en la interfaz de usuario, que al presionarlo comunicará al modelo que remueva la referencia a la instancia de usuario que estaba usando el sistema, y por lo tanto se regresará al estado inicial de la aplicación.

Cabe destacar, que los algoritmos de seguridad son triviales, dado que esta tesis no tiene nada que aportar en este aspecto, aunque podría cambiarse por algún otro algoritmo que encriptara los datos de alguna forma, sí así se requiriera en algún momento.

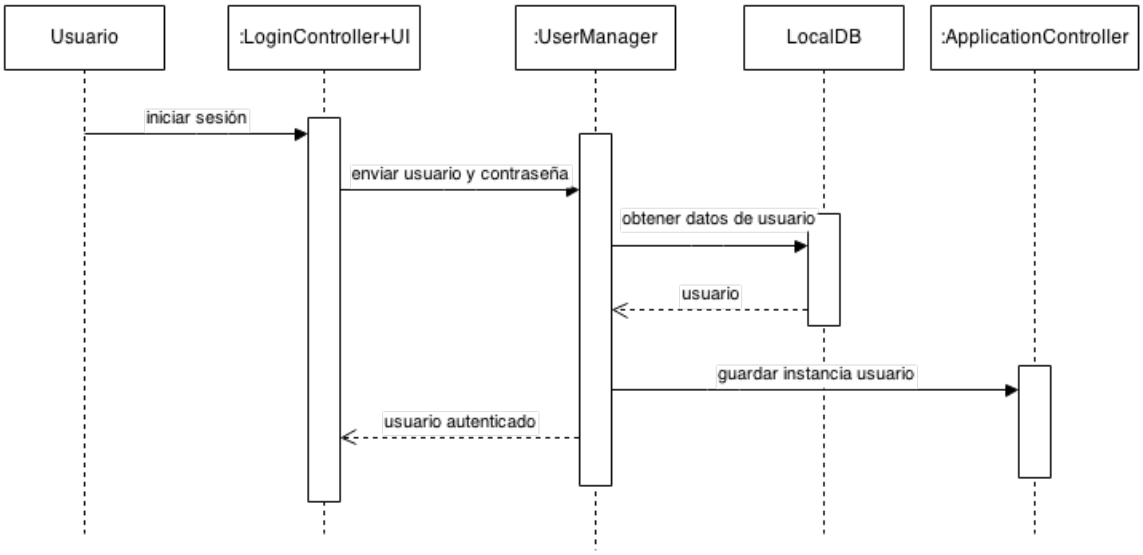

Figura 3.11: Diagrama de secuencia para iniciar sesión.

3.3.2. Administración de proyectos

Mediante este módulo, el usuario podra crear proyectos y visualizar los proyectos que ya había creado anteriormente. Ya que pueden existir diferentes usuarios, a cada proyecto se le asigna quien es su creador o responsable, por lo que una persona no podrá ver ni manipular los proyectos de otra persona, en el modo individual del sistema. La manipulación del proyecto de otra persona, sólo se lleva a cabo en el modo colaborativo.

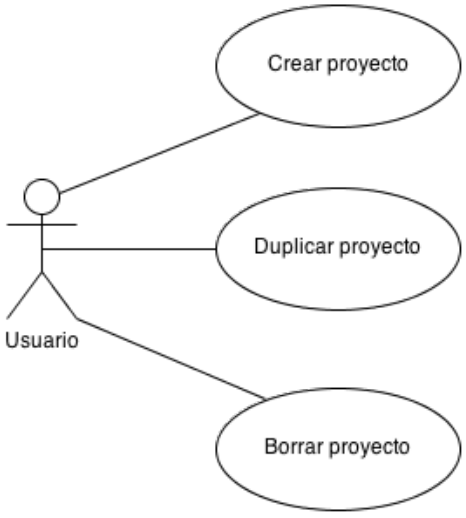

Figura 3.12: Caso de uso de la administración de proyectos.

Existen tres tareas principales para la administración de proyectos, las cuales se muestran en la Figura 3.12.

Los proyectos se guardan en la base de datos. La información que se requiere en la tabla de proyectos se muestra en la Figura 3.13.

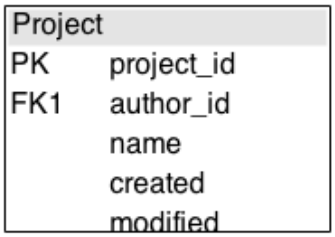

Figura 3.13: Tabla de base de datos para proyectos.

Al momento de crear un proyecto, solo necesitamos un nombre o identificador que se le quiera dar. Los demás datos, como el usuario autor del proyecto y las fechas de creación y modificación, se calculan solas en base al usuario que esta usando la aplicación y la fecha actual, respectivamente. La Figura 3.14 muestra el diagrama de secuencia para la creación de un proyecto nuevo. Se debe ingresar un nombre de proyecto y después solicitar su creación. *ProjectController* pasa los datos del nuevo proyecto a *ProjectManager*, que se encarga de guardar el nuevo proyecto en la base de datos. Al final se visualiza el nuevo proyecto en *ProjectUI*, que muestra un listado de los proyectos del usuario que esta usando el sistema.

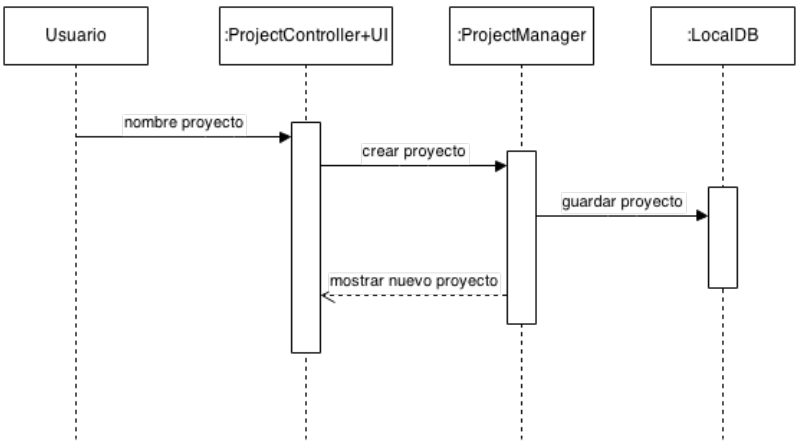

Figura 3.14: Diagrama de secuencia para crear un proyecto.

El mecanismo para borrar un proyecto es una función sencilla, ya que se presentará en *ProjectUI* un listado de los proyectos que existen, donde el usuario podrá elegir uno para manipularlo, y una de las acciones es la de borrar el proyecto. Borrarlo hará que se elimine el proyecto de la base de datos.

3.3.3. Área de trabajo y herramientas de dibujo

El área de trabajo es el módulo más importante de la aplicación, ya que es donde se visualiza todo el trabajo que los usuarios hacen para la creación de su mapa mental. El mapa mental es parte de un proyecto, por lo que este último debe de existir antes de poder trabajar en un mapa mental.

Mediante las herramientas de dibujo, el usuario agrega, modifica y remueve nodos que componen el mapa mental. En una sesión colaborativa, cada usuario tendrá su propia herramienta de dibujo, manipulando el mapa mental que se muestra en el área de trabajo que se ha redistribuido entre los diferentes dispositivos.

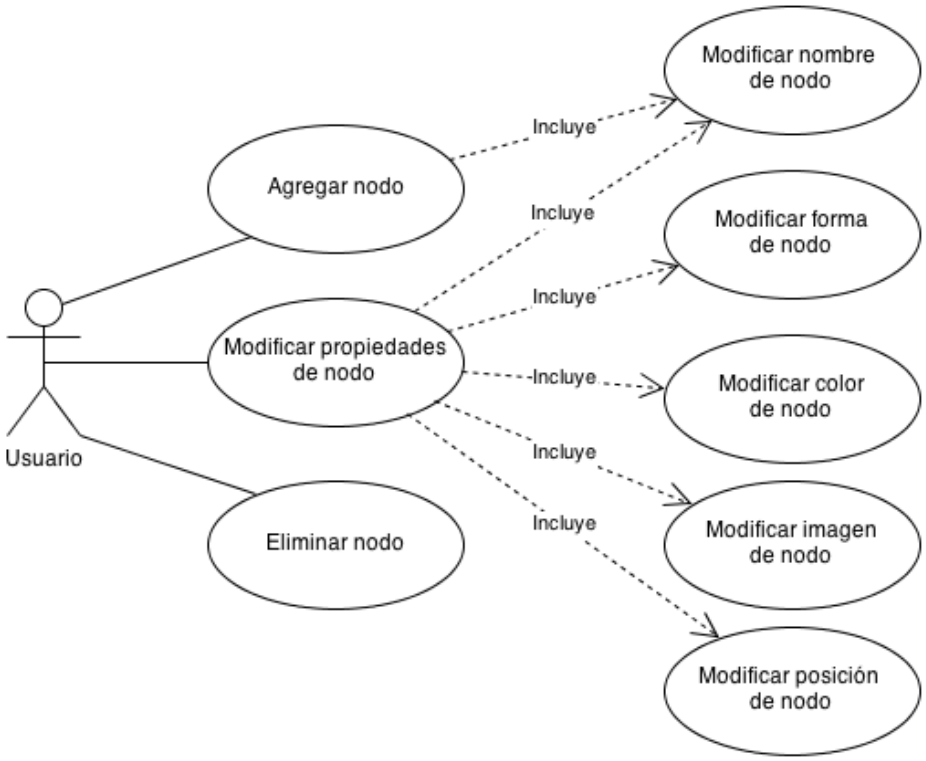

Figura 3.15: Caso de uso para el área de trabajo y herramientas de dibujo.

En la Figura 3.15 se muestran las principales tareas que se pueden llevar a cabo para interactuar con el área de dibujo, y estas tareas se realizan mediante las herramientas de dibujo.

Se le llama “nodo” a una elementro dentro del grafo que se genera del mapa mental. El grafo usado para expresar los mapas mentales es un grafo no dirigido, ya que no importa la dirección en la cuál se crean aristas para relacionar nodos entre sí. Los nodos dentro del grafo no necesariamente deben estar conectados con otro nodo, es un grafo muy libre que permite al usuario manipularlo como él lo deseé.

El usuario puede manipular propiedades del nodo, como el texto y el color, incluir una imágen dentro del nodo, y cambiar la posición del nodo dentro de toda el área de trabajo.

El área de trabajo está compuesta por un lienzo en el cuál el usuario puede agregar nodos y conectarlos entre sí. La Figura 3.16 muestra como se visualiza el área de trabajo cuando el usuario entra por primera vez a un proyecto nuevo. El sistema de coordenadas utilizado en el área de trabajo tiene su origen en la esquina superior izquierda, por lo que se aumenta el eje X hacia la derecha, y se aumenta el eje Y hacia abajo.

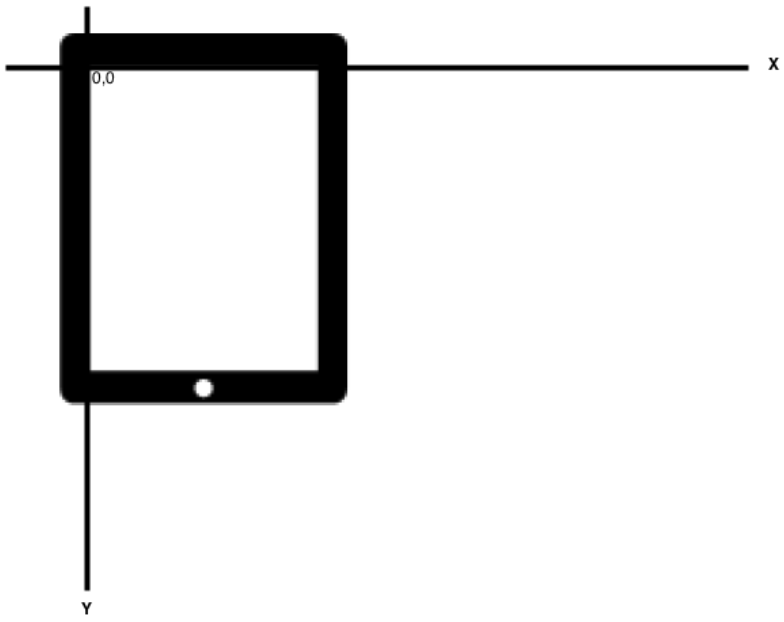

Figura 3.16: Lienzo de área de trabajo.

El usuario podrá desplazarse por el área de trabajo, usando gestos de arrastre que permitirán mover el lienzo de una forma natural. Así que el lienzo podrá crecer tanto como el usuario lo requiera.

3.4. Modo colaborativo

El modo colaborativo existe cuando un usuario requiere compartir un mapa mental con un colaborador. La colaboración se logra ejecutando una serie de pasos. Para que el modo funcione, los usuarios deben de estar conectados a la misma red local, para que puedan verse el uno al otro.

La tecnología bluetooth permite el intercambio de datos entre dispositivos que cuenten con los sensores necesarios, sin embargo, los dispositivos móviles que existen en el mercado tienen diferentes características y cuentan con diferentes capacidades. Con esta tecnología, se podrían conectar hasta 8 dispositivos, uno de ellos actuando como un servidor que coordina la comunicación, pero los dispositivos que se encuentran en el mercado, llegan a soportar una conexión con sólo otro dispositivo, y cuando se conectan con más de 1, la conexión ya no es muy confiable, llegando a perder el enlace con facilidad [Madhavapeddy and Tse, 2005]. Existen nuevas tecnologías como Wi-Fi Direct, que es parecida a bluetooth en cuanto al intercambio de datos a corta distancia, aunque es más poderosa. Pero al ser tecnologías innovadoras, no todos los dispositivos cuentan con lo necesario para usarlas, por lo tanto no es buena opción para usar en un proyecto real aún.

Este trabajo de tesis optó por elegir una topología cliente-servidor, ya que es una forma de trabajar que todos los dispositivos modernos pueden utilizar, siempre y cuando se encuentren dentro de la misma red local (LAN).

3.4.1. Pasos para iniciar colaboración

Para iniciar una colaboración entre varios usuarios, se deben de llevar a cabo una secuencia de pasos. La Figura 3.17 muestra el diagrama de actividad que se usa para transicionar del modo individual al colaborativo, y viceversa.

Los pasos mostrados en la Figura 3.17 se realizan entre el cliente y el servidor, es un proceso de comunicación que al final hace que cada dispositivo ejecute tanto la redistribución como la remodelación de su interfaz gráfica de usuario. A continuación se explica que sucede en cada uno de los pasos.

1. **Hacer gesto para iniciar colaboración**: En el primer paso se debe realizar el gesto de arrastre para iniciar la colaboración con otro dispositivo. Los gestos son descritos a más detalle en la subsección 3.4.2. La forma natural de llevar a cabo este proceso, es posicionar un dispositivo al lado de otro, y que cada persona que lo opera haga un gesto de arrastre hacia la dirección en donde se encuentra el otro dispositivo. Los dos gestos deben de realizarse en menos de un intervalo de tiempo que se especifíca en el sistema, por ejemplo, 1 segundo.

2. **Reconocimiento de gestos**: Al realizar los gestos en sincronía, el sistema será capaz de reconocer que se desea iniciar una conexión, por lo que el dispositivo que actúa como servidor iniciará el proceso de colaboración. Cuando se inicia la colaboración entre usuarios que no se encontraban colaborando en ese momento con nadie más, entonces uno de los dispositivos de los usuarios ac-

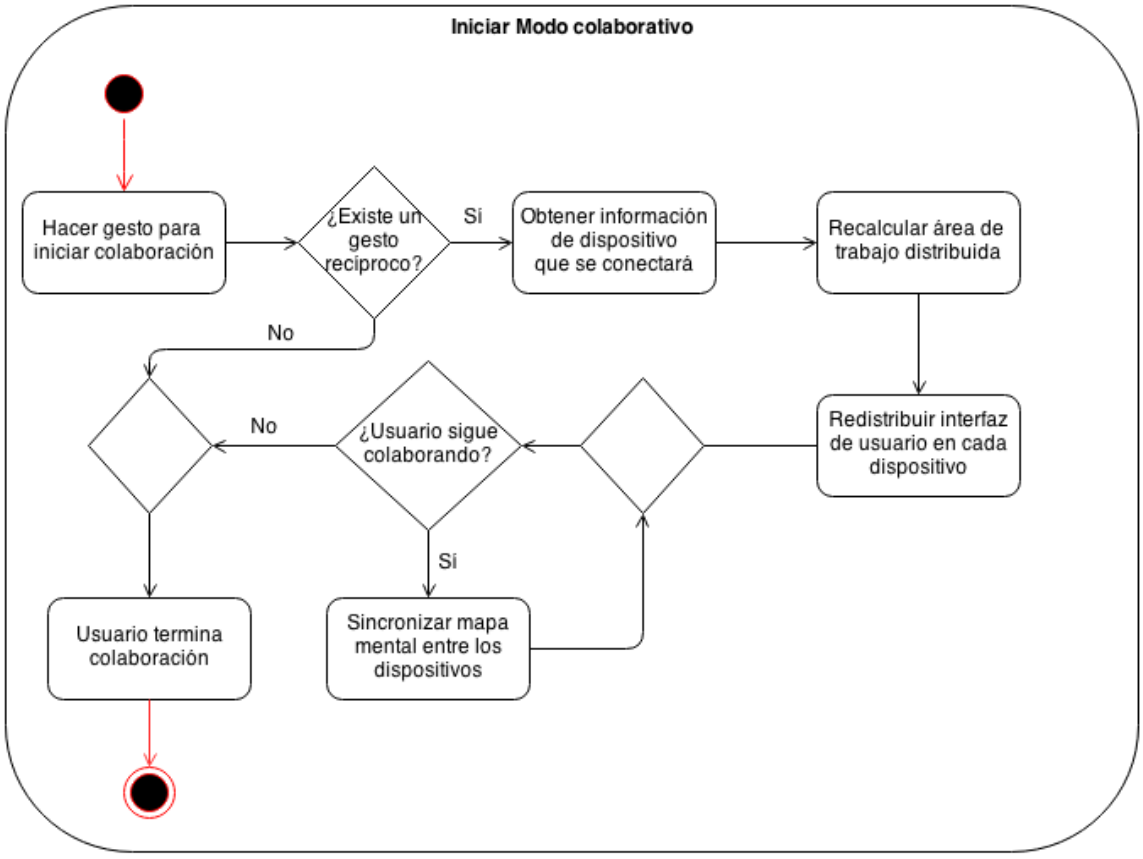

Figura 3.17: Diagrama de actividad para iniciar colaboración.

tuará como servidor coordinador por el resto de la colaboración, aunque lleguen más dispositivos a unirse. Esta comunicación se hace desde el *NetworkManager* del cliente hacia la clase *CollaborativeCoordinator* en el dispositivo que actúa como servidor coordinador.

3. **Obtención de información de dispositivo**: Cada dispositivo enviará al servidor sus datos cuando acaba de iniciar la sesión colaborativa. Esta información incluye un identificador único del dispositivo, y su resolución de pantalla, así como el usuario que esta usando la aplicación en ese dispositivo. La información del cliente se envía desde *NetworkManager* hacia *CollaborativeCoordinator*, y este último guarda esta información en *CollaborativeProject*, donde se crea una instancia *CollaborativeUser* por cada usuario que participa en la colaboración.

4. **Recalcular área de trabajo**: Se recalcula el área de trabajo colaborativa, tomando en cuenta que se ha unido un nuevo usuario. Se asignará a cada dispositivo el área que estará visualizando en su versión de mapa mental. El pseudocódigo que realiza la redistribución es mostrado en el Algoritmo 2. Los

mecanismos de redistribución se llevan a cabo por *MentalMapManager*.

5. **Redistribución de interfaz de usuario**: Se notifica a cada dispositivo el área de trabajo que estará visualizando.

6. **Sincronización de mapa mental**: Mientras que los usuarios esten trabajando sobre el mapa mental en la sesión colaborativa, cada cambio que hagan al mapa será notificado a los demás clientes, para que todos tengan el mismo estado del mapa. La comunicación se lleva a cabo entre *NetworkManager* en el cliente, y *CollaborativeCoordinator* en el servidor coordinador. Dentro del servidor se sincroniza el mapa mental usando *CollaborativeProject* y *MentalMapManager*.

7. **Desconexión**: Una vez que un usuario requiera finalizar su trabajo colaborativo, puede retirarse. Aquí también entra el caso en el que el usuario pierda la conexión con el sistema. Cuando esto sucede, el sistema nuevamente repite los pasos 4 y 5, para redistribuir la interfaz entre los dispositivos que siguen interactuando, hasta que ningún dispositivo quede en la interacción y entonces todos pasen de nuevo al modo individual.

3.4.2. Redistribución de interfaz de usuario

La redistribución de la interfaz gráfica de usuario, es un proceso que se ejecuta en el servidor, por lo tanto aquí se coordina el área de trabajo colaborativa.

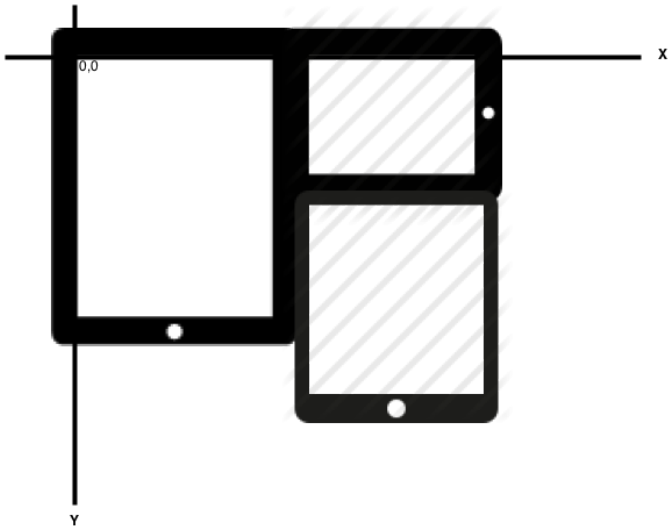

Figura 3.18: Ejemplo de redistribución en modo colaborativo.

La Figura 3.18 muestra un ejemplo de como podría estar organizada un área de

trabajo entre tres dispositivos. Cada uno de ellos podrá visualizar el área que le corresponde. Además se podrán intercambiar los nodos del mapa mental, ya que es la manera natural en la que se trabajaría. La forma de pasar los elementos es una extensión a lo que se ha realizado en trabajos como Pick-and-Drop [Rekimoto, 1997, Rekimoto and Saitoh, 1999].

El proceso de redistribución inicia por parte del cliente que desea colaborar con alguien más. Uno de los clientes que interactúan en el sistema, pasa a ser el servidor que coordinará la comunicación en la sesión colaborativa. Cada cliente que quiere conectarse debe hacer un gesto de conexión. Este gesto se realiza dentro del área de trabajo y debe de realizarse como un arrastre hacia 1 de 4 direcciones posibles. Un usuario puede conectarse con otro dispositivo que se encuentre arriba, a la derecha, a la izquierda, o abajo del suyo.

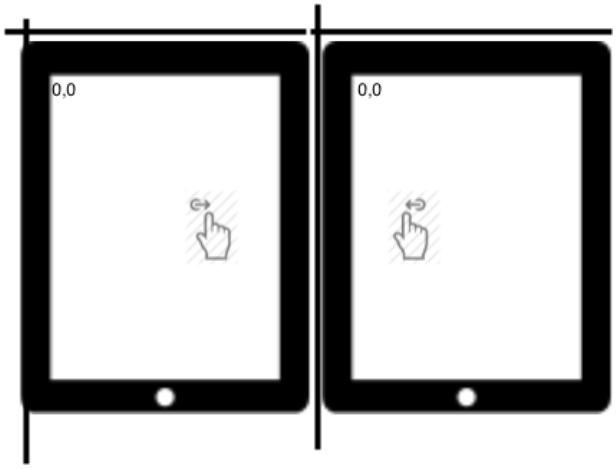

Figura 3.19: Gestos para iniciar colaboración.

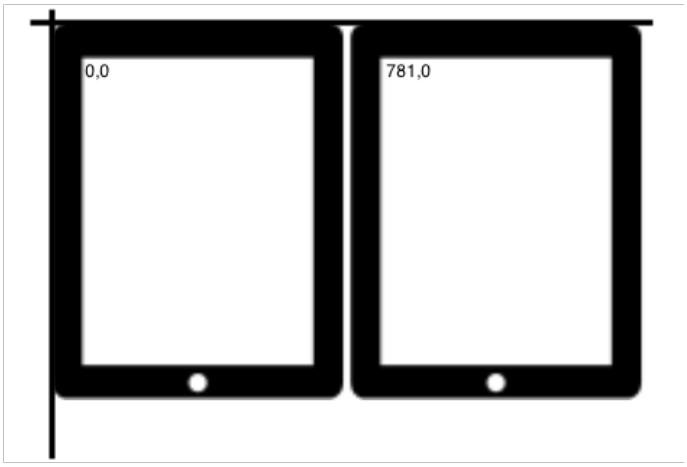

Figura 3.20: Dispositivos despues de iniciar colaboración.

En la Figura 3.19 se ejemplifica como se inicializa un proceso de colaboración entre dos dispositivos. En este ejemplo se posicionan dos dispositivos uno al lado de otro, y ambos están en su modo individual, ya que como se puede ver en la primera parte de la figura, ambos tienen sus coordenadas en el origen $(0, 0)$.

Ambos usuarios hacen un gesto de arrastre hacia el dispositivo que desean conectarse, lo cual nos lleva a la Figura 3.20, ya que ahora solo un dispositivo mantuvo su posición original y, en este caso el de la derecha se ha recorrido dentro del área de trabajo, que ahora está compartida entre ambos dispositivos. Suponiendo que los dispositivos tienen una resolución de 780x1024, entonces el dispositivo de la derecha se recorre 780 pixeles hacia la derecha, para desplegar la sección del lienzo que le corresponde. Una vez unidos estos dispositivos, pueden unirse más, tal como se muestra en la Figura 3.18.

El pseudocódigo que logra la conexión y redistribución, es el que se muestra en los algoritmos 1 y 2.

Algorithm 1 Algoritmo para iniciar conexión

```
function connectToCollaborate(device)
    if !Q.isEmpty() then                    ▷ Q es la cola de conexiones pendientes
        otherDevice ← Q.peek()
        if device.isCompatibleWith(otherDevice) then
            Q.poll()
            connectDevices(device, otherDevice)        ▷ Este método recalcula las
coordenas (Algoritmo 2)
        end if
    end if
    Q.add(device)
end function
```

En el algoritmo 1, un dispositivo trata de iniciar una nueva conexión colaborativa, mandando esta señal al servidor. Para que se realize una conexión exitosa, el servidor mantiene una cola de conexiones pendientes, que contiene todos los intentos de conexiones que se han intentado hacer hasta ese momento. Cada intento de conexión tiene un temporizador asociado, por ejemplo de 1 segundo, y si en este lapso de tiempo no llega una conexión compatible, entonces este intento se remueve automáticamente de la cola, gracias al temporizador. Si se diera un intento de conexión compatible, quiere decir que se hizo el gesto sincronizado desde dos dispositivos.

Para poder redistribuir la interfaz de usuario, se usa el algoritmo 2. En éste, los dispositivos son agregados a la cola de conexiones activas y se supone que el dispositivo B es el que se quiere conectar a una sesión colaborativa con el dispositivo A.

Se pueden conectar haciendo gestos hacia 4 direcciones diferentes, que son conectarse

Algorithm 2 Algoritmo para conectar dispositivos y redistribuir interfaz

```
function connectDevices(deviceA, deviceB) ▷ El dispositivo B se incorpora al espacio de trabajo
    Q.add(deviceA)                        ▷ Q es la cola de conexiones activas
    Q.add(deviceB)
    if deviceA.isDown() then
        deviceB.setViewX(deviceA.getViewX() − deviceB.getViewX())
        deviceB.setViewY(−deviceB.getHeight())
    else if deviceA.isLeft() then
        deviceB.setViewX(deviceA.getWidth())
        deviceB.setViewY(deviceA.getViewY() − deviceB.getViewY())
    else if deviceA.isTop() then
        deviceB.setViewX(deviceA.getViewX() − deviceB.getViewX())
        deviceB.setViewY(deviceA.getHeight())
    else if deviceA.isRight() then
        deviceB.setViewX(−deviceB.getWidth())
        deviceB.setViewY(deviceA.getViewY() − deviceB.getViewY())
    end if
    server.sendNewCoordinatesToDevice(deviceB.getId(), deviceB)
end function
```

hacia un dispositivo arriba, abajo, a la izquierda, o a la derecha. Por lo tanto, se compara en que posición está el dispositivo A, con base en la posición física que tiene el dispositivo B. Entonces, se puede recalcular la porción de lienzo que el dispositivo B puede ver, y se envía al dispositivo B las nuevas coordenadas en las que su vista debe posicionarse para que haga el efecto de que los dispositivos están compartiendo la misma área de trabajo.

Manipulación de área de trabajo compartida

Una vez que se ha entrado completamente al modo de colaboración, los usuarios pueden trabajar en sincronía, manipulando el mapa mental juntos.

Cualquier usuario puede agregar un nodo al mapa mental en cualquiera de los dispositivos, ya que cada uno cuenta con sus herramientas de dibujo. Esta etapa funciona tal y como si estuvieran trabajando de manera individual, pero se podrá apreciar al mismo tiempo qué es lo que están haciendo los otros compañeros, ya que todos se encuentran co-localizados y las pantallas de los dispositivos están visibles para todos.

Dentro de una sesión colaborativa, existen otros gestos para enviar un nodo del mapa mental de un dispositivo a otro. El primer gesto se llama *drag* y se ejemplifica en la

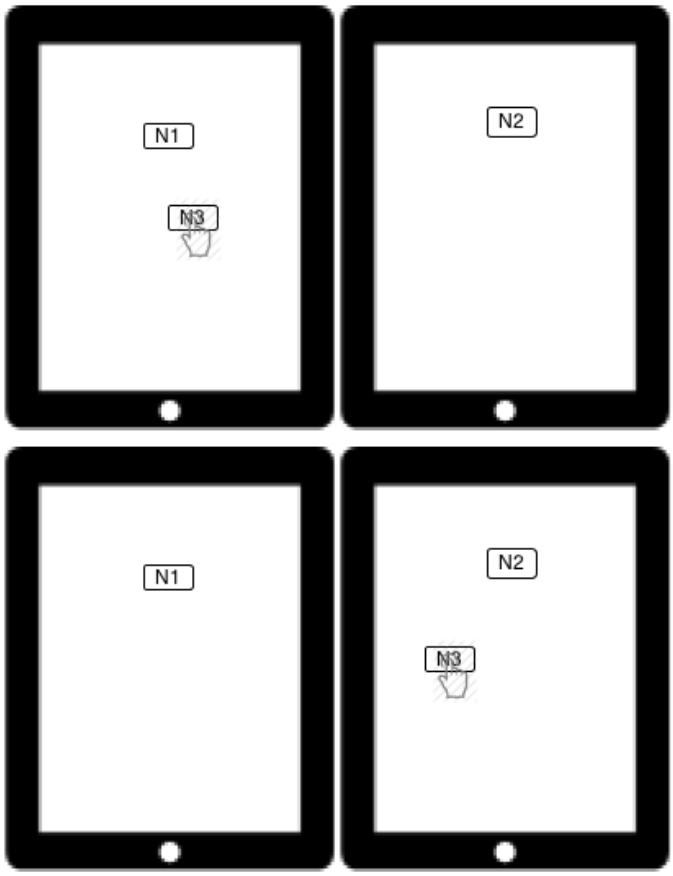

Figura 3.21: Gesto "drag"para compartir un nodo con otro dispositivo.

Figura 3.21. En este se muestra como se puede pasar un elemento de una pantalla a otra, simplemente uno continúa haciendo un gesto de arrastre de un nodo, traspasando los límites de la pantalla en la que se encuentra el nodo actual y siguiendo el gesto dentro de la pantalla de uno de los dispositivos vecinos.

El otro gesto, llamado *fling*, es una alternativa para pasar el nodo de una manera más rápida, cuando no se quiere pasar manualmente el nodo hasta la posición deseada. Usando un gesto de arrastre y después se suelta el dedo, como sí se tratara de lanzar el nodo, entonces éste se mostrará ahora en el dispositivo vecino. El ejemplo es mostrado en la Figura 3.22, que usa los mismos nodos de la figura del gesto de *drag*, pero después de hacer el gesto de *fling*, el nodo se posiciona en el centro de la pantalla del dispositivo destino. El efecto será que el nodo desaparece en un dispositivo y aparece en otro.

En el diagrama de clases general (Figura 3.5) se mostraron las instancias que mantienen la comunicación entre el cliente y el servidor. Para que todos los clientes estén en sincronía con el estado actual del mapa mental, el *CollaborativeCoordinator* en el servidor, notifica a cada una de las instancias *NetworkManager* que estén en la misma sesión colaborativa de los cambios que realizó cualquiera de los clientes que

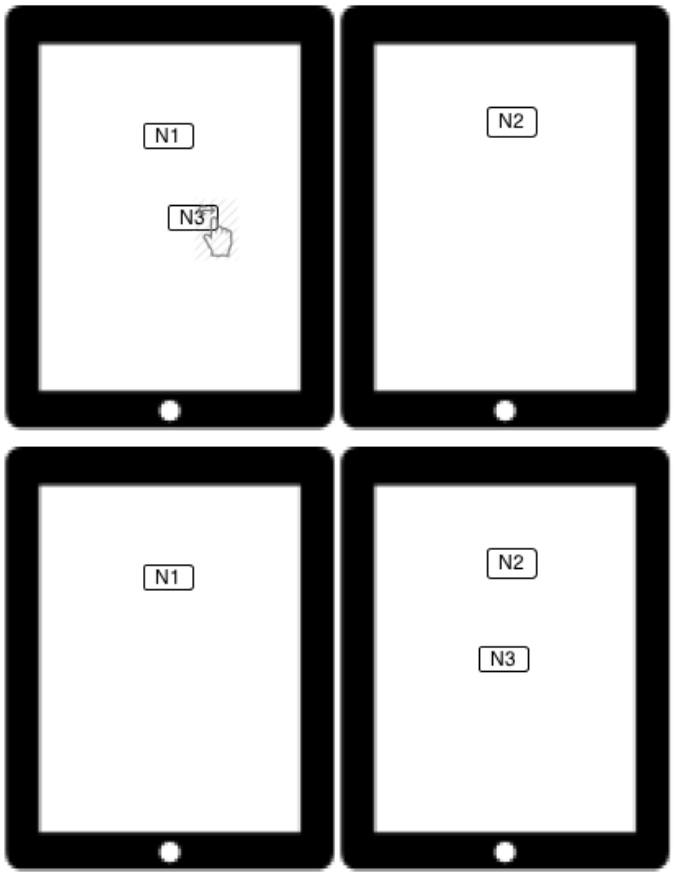

Figura 3.22: Gesto "fling"para compartir un nodo con otro dispositivo.

estén comunicandose entre sí. Los cambios se persisten conjuntamente entre *CollaborativeProject* y *MentalMapManager*.

Capítulo 4

Implementación

El sistema de mapas mentales es implementado en una de las plataformas móviles más populares que existen en la actualidad. Se le dio el nombre Flowlure al sistema de mapas mentales, para darle una identidad y pueda ser reconocido fácilmente. En este capítulo se da a conocer la plataforma móvil usada para implementar el sistema, esta plataforma es Android (sección 4.1). Después se habla de como se ha implementado la interfaz de usuario (sección 4.2). Se pasa a revisar cada uno de los módulos del sistema, incluyendo la administración de usuarios, la administración de proyectos y el área de trabajo (sección 4.3). También se explica como se logra la conexión entre clientes y la sincronización de los mensajes entre los mismos, logrando la redistribución de la interfaz de usuario (sección 4.4).

4.1. Tecnología utilizada

La aplicación de los mapas mentales debe de ejecutarse en dispositivos móviles, porque actualmente estos dispositivos son los más usados ya que facilitan las tareas a las personas en cualquier lugar en donde se encuentren, como por ejemplo el compartir información con algún conocido. Sin embargo, existen muchos dispositivos que varian en tamaño, precio y las características de hardware que ofrecen. Por lo tanto se decidió elegir una de las plataformas móviles más utilizadas en el mercado. A continuación se da explica cuál es y qué tipo de dispositivos se utilizaron.

4.1.1. Plataforma Android

Android es un sistema operativo que está basado en Linux, pero que tiene un enfoque en dispositivos móviles. Se eligió trabajar sobre este sistema operativo porque 1) es gratuito, 2) es distribuido bajo licencia de software libre, 3) en los últimos años ha ganado una enorme popularidad que va aumentando con el paso del tiempo.

Android es más que sólo el sistema operativo, porque brinda a los desarrolladores herramientas por medio de las cuales pueden crear sus aplicaciones para esta este sistema operativo. Por lo tanto, la plataforma Android ofrece una API (del inglés Application Programming Interface, que traducido es Interfaz de Programación de Aplicaciones) por medio de la cual el desarrollador puede comunicarse con el sistema operativo y hacer todo tipo de aplicaciones.

La plataforma está desarrollada en su mayor parte en el lenguaje de programación C, sin embargo, los desarrolladores pueden usar un lenguaje orientado a objetos para codificar, el cual es Java. Esto es porque las APIs y bibliotecas ofrecidas fueron desarrolladas en Java, lo que libera al programador de trabajar en un lenguaje de más bajo nivel y que no sea orientado a objetos.

Usar un lenguaje de programación como Java, es perfecto para el desarrollo de este trabajo de tesis, porque todo lo que se discutió en el capítulo 3, es fácilmente aplicado mediante un lenguaje de programación de este tipo. Aunque si fuera necesario, también se puede usar el lenguaje de programación C, para hacer llamadas directas al sistema operativo, ignorando la API que ofrece Android en Java. Lo último sólo se llevaría a cabo si no fuera suficiente lo que ofrece la API, pero no es el caso para la aplicación de mapas mentales.

Entonces para trabajar sobre la plataforma Android, es necesario descargar todas las herramientas de programador en la máquina que se va a trabajar, que puede ser una laptop o PC. Se puede encontrar la última versión de la plataforma en el sitio web de Android.

También se necesita de un entorno de desarrollo (IDE), mediante el cual el programador deberá de codificar toda la aplicación. El IDE elegido es Eclipse ya que se combina fácilmente con las herramientas de programador de Android.

4.1.2. Dispositivos Android

Actualmente, es muy difícil saber el número exacto de dispositivos que funcionan bajo la plataforma Android. El tipo de dispositivos que usan Android van desde

smartphones, tabletas, y dispositivos embebidos.

Para la realización del sistema de mapas mentales se optó por utilizar smartphones y tabletas, que son los más usados por las personas. También es importante saber que existen diferentes versiones de Android, ya que la plataforma va mejorando, ofreciendo características nuevas que algunos dispositivos más antiguos no pueden tener. Por ejemplo, sólo los más nuevos dispositivos cuentan con tecnología Wi-Fi Direct, que es una tecnología de comunicación peer-to-peer parecida a Bluetooth, así que los dispositivos más antiguos no pueden hacer uso de esta tecnología.

Más del 95 % de dispositivos que existen en el mercado tienen al menos la versión 2.1 de Android, por lo tanto el sistema de mapas mentales está desarrollada para soportar todos estos dispositivos que tengan una versión igual o mayor a la 2.1.

4.2. Interfaz de usuario

La aplicación de mapas mentales cuenta con dos tipos de interfaz de usuario:

1. *Interfaz estática*. Es el tipo de interfaz de usuario que existe antes de la ejecución de la aplicación. De antemano se conoce el dispositivo que la usará.

2. *Interfaz dinámica*. Este tipo de interfaz de usuario organiza los componentes que se muestran en pantalla en el tiempo de ejecución. Así que se podrán obtener datos del dispositivo, como el tamaño de su pantalla para saber en donde posicionar los componentes, entonces la interfaz de usuario se verá diferente dependiendo del dispositivo.

Este tipo de interfaces son importantes para implementar la remodelación y redistribución plástica que soportará el sistema, ya que la aplicación funciona de forma indivual y colaborativa, usando smartphones y tabletas.

En el capítulo 3 se identificaron los tres módulos más importantes para el sistema de mapas mentales, los cuales son 1) manejo de usuarios, 2) administración de proyectos, 3) área de trabajo y herramientas de dibujo. Cada uno de estos módulos es mostrado en una sección de la interfaz de usuario.

El módulo de manejo de usuarios cuenta con su propia interfaz, aislada de las demás. Por otra parte, el módulo de administración de proyectos y del área de trabajo, aunque cuentan con una interfaz única, ambas se visualizan en la misma pantalla en una área diferente.

A continuación se muestra como está organizada la interfaz de usuario de toda la aplicación.

4.2.1. Interfaz para manejo de usuarios

La primera pantalla que se presenta en el sistema permite al usuario iniciar sesión o crear una cuenta nueva, sí aún no tiene una.

La Figura 4.1 muestra la pantalla inicial de la aplicación, en donde el usuario puede ingresar un nombre de usuario y contraseña para ingresar en el sistema. Así mismo, existe un botón para iniciar sesión en el sistema o para registrar a un nuevo usuario.

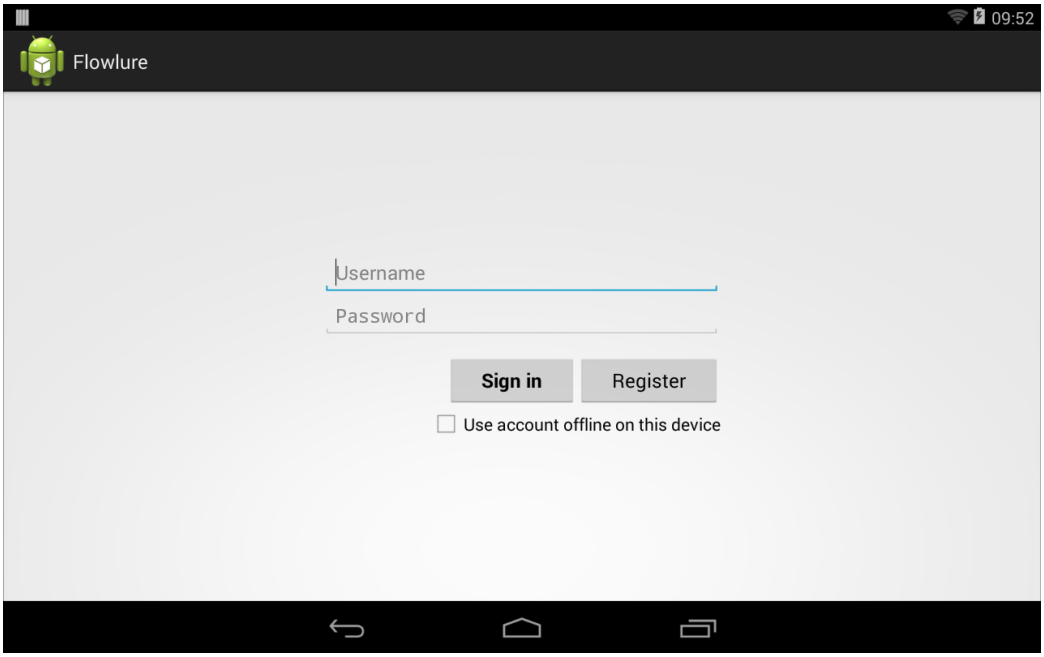

Figura 4.1: Pantalla inicial de la aplicación.

Ambos botónes usan los campos de nombre de usuario y contraseña, pero tienen diferente funcionalidad. Es decir, el botón de inicio de sesión comunica al sistema que busque un usuario existente, mientras que el botón de registro verifica que no exista un usuario con el mismo nombre para después almacenar los datos del nuevo usuario en la aplicación y que pueda iniciar sesión de aquí en adelante.

Otro elemento importante es el checkbox que aparece debajo de los botónes. Con este componente se indica al sistema que cuando se inicie sesión con un usuario, él usuario se almacene localmente en el dispositivo. Más adelante se explicará como se almacenan los datos en el sistema.

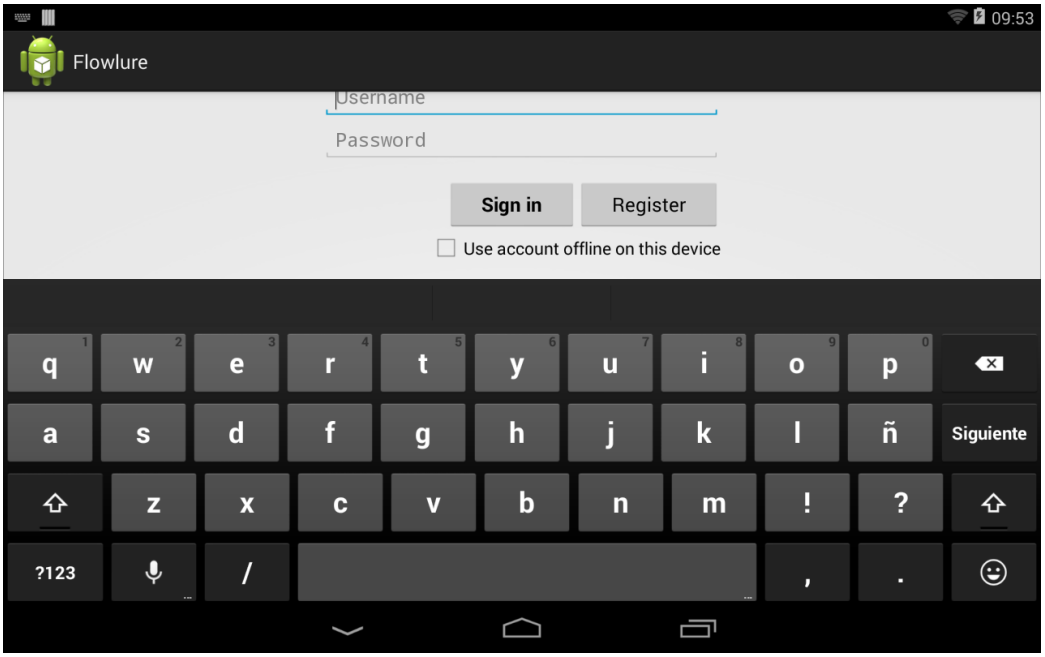

Figura 4.2: Pantalla inicial de la aplicación con teclado visible.

La Figura 4.2 muestra la misma pantalla de inicio de sesión, pero mostrando un teclado virtual que el usuario debe usar para ingresar datos dentro de la aplicación. El teclado virtual es parte de Android, y se muestra siempre que se pidan datos al usuario.

4.2.2. Interfaz principal

Se le llama interfaz de usuario principal a la pantalla que encapsula al los módulos de administración de proyectos y el área de trabajo. La Figura 4.4 muestra la interfaz de usuario principal cuando el usuario acaba de iniciar sesión en el sistema.

Está interfaz de usuario esta dividida en dos secciones: 1) la parte de la izquierda muestra lo relacionado al módulo de administración de proyectos, y 2) es la parte de la derecha o central de la interfaz, que muestra un lienzo blanco en donde el usuario puede manipular su mapa mental, así que éste último maneja el módulo del área de trabajo.

Se decidió juntar ambos componentes en una sola pantalla porque es más eficiente para el usuario. Una persona puede usar la aplicación y rápidamente cambiar entre proyectos usando el panel de proyectos que se muestra en la sección de la izquierda de la Figura 4.3 y al elegir el proyecto puede trabajar de manera inmersa sobre el mapa mental. Cuando se ha seleccionado un proyecto o simplemente se cierra el panel de

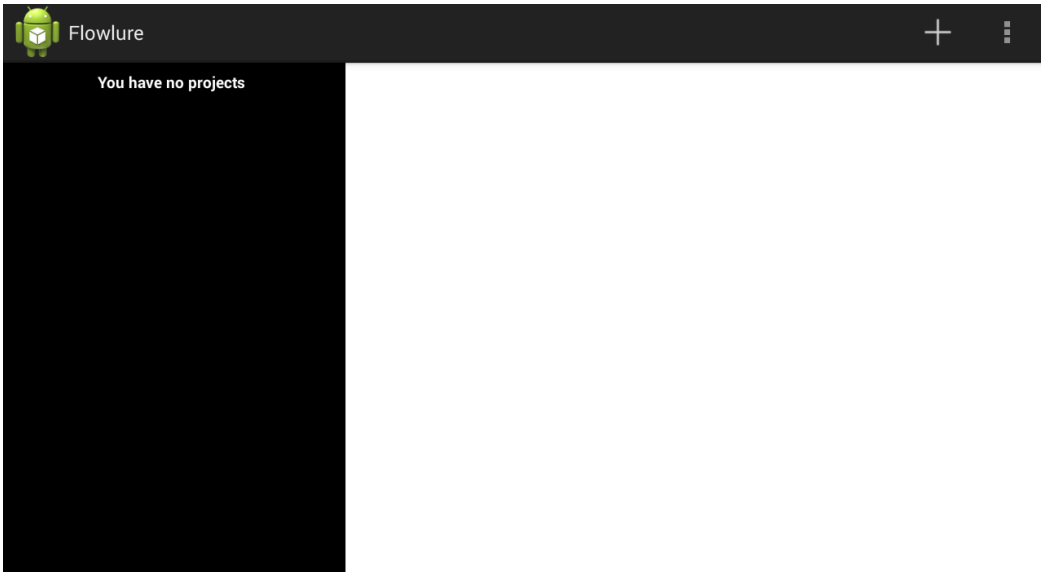

Figura 4.3: Pantalla principal sin proyectos.

proyectos, el usuario puede usar el área de trabajo para manipular su mapa mental.

Figura 4.4: Pantalla de área de trabajo vacía.

La Figura 4.4 muestra el lienzo para el área de trabajo de trabajo en donde el usuario puede manipular el mapa mental. La Figura 4.5 muestra un ejemplo de como se vería un mapa mental después de que el usuario ya lo ha manipulado, más adelante se epxlicará cómo se implementa esta funcionalidad.

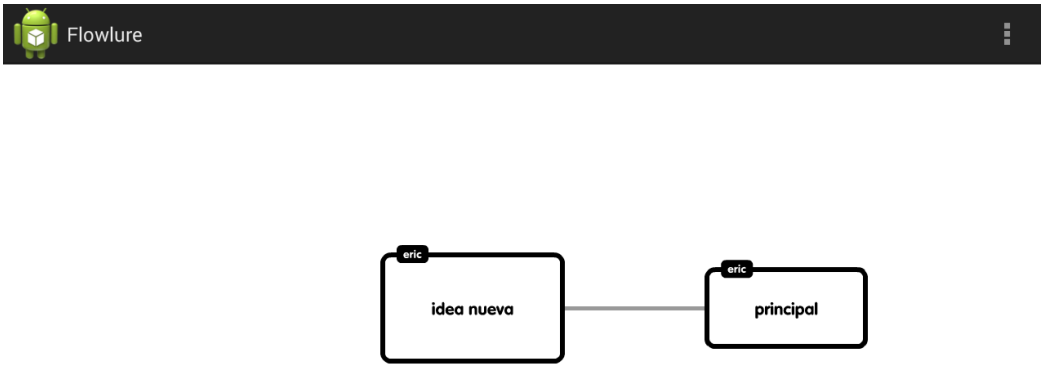

Figura 4.5: Ejemplo de mapa mental iniciado.

4.3. Implementación de módulos

En el capítulo 3 se dieron a conocer los módulos más importantes del sistema de mapas mentales. Los módulos incluyen la administración de usuarios, la administración de los proyectos, el área de trabajo y herramientas de dibujo, que en este caso será visto como la manipulación del mapa mental.

4.3.1. Implementación de módulo de usuarios

El módulo de usuarios es importante porque se requiere para que una persona pueda hacer uso de la aplicación. Se necesita que el usuario inicie sesión para identificarlo dentro de una sesión colaborativa de mapa mental, si no se hiciera de esta manera, no se podría identificar a la persona que esta haciendo algún cambio al mapa mental.

Mediante éste módulo se crean usuarios, se inicia sesión en el sistema y también se cierra sesión. Por lo tanto, se necesita de una base de datos en la que se pueda guardar la información de cada usuario. Android cuenta con una base de datos por defecto, conocida como SQLite. Es una versión básica de cualquier motor de base de datos que se pueda encontrar para una PC, ya que toda la base de datos se guarda en un archivo privado dentro del dispositivo. Android es suficientemente robusto para manejar la seguridad de acceso a la base de datos, ya que no permitirá que aplicaciones diferentes al mapa mental, traten de manipular la base de datos de esta aplicación.

SQLite obedece la sintáxis SQL, por lo que se puede crear una tabla con dos columnas: una para el nombre del usuario y otra para la contraseña. El nombre de usuario será la clave primaria dentro de la tabla, lo que no permitirá que existan nombres de usuario duplicados.

4.3.2. Implementación de módulo de proyectos

Un usuario podrá crear todos los proyectos que requiera. Cada proyecto tendrá asociado un archivo que tendrá toda la información referente al mismo, incluyendo quien es el creador y todos los datos sobre el mapa mental.

No se maneja una base de datos para el proyecto, porque se contempló que esta información deberá de ser compartida, ya que un usuario comparte su proyecto en una sesión colaborativa. Por lo tanto, sí se usará una base de datos, se tendrían que hacer varias peticiones a la base de datos, y traspasar toda esta información por un medio de comunicación al dispositivo con el que se desea compartir el proyecto.

Al usar un archivo es más fácil compartir el proyecto, ya que únicamente se transferiría el archivo al dispositivo con el cuál se desea compartir. Este archivo seguirá la sintáxis de JSON, ya que es un formato establecido y que sigue reglas muy simples. Un archivo JSON funciona como un diccionario de datos, en donde cada valor tiene asociada una llave única, por lo que si un usuario conoce la llave podrá obtener el valor.

4.4. Implementación de redistribución

Para lograr la redistribución entre varios dispositivos heterogéneos, los dispositivos deben contar con capacidad para comunicarse entre ellos. En el capítulo 3 se vió que el sistema usa una topología de comunicación cliente-servidor, por medio de la cual todos los dispositivos se comunicarán a un único servidor, quién coordinara la comunicación entre los dispositivos.

Los dispositivos deben de contar con comunicación inalámbrica usando la tecnología Wi-Fi, por medio de la cual podrán lograr la comunicación cliente-servidor. Por lo tanto, todos deben de pertenecer a la misma red de área local (LAN).

4.4.1. Biblioteca de comunicación

Se usó la biblioteca de software libre Kryonet para implementar la comunicación entre los clientes y el servidor. Se eligió esta biblioteca porque es gratuita y está desarrollada en Java.

Una característica importante de la biblioteca, es el intercambio de objetos serealizados, es decir, puedes usar una instancia de un objeto que tengas en tu cliente y pasar esta instancia al servidor, usando la API que brinda Kryonet.

A fin de cuentas se usan sockets TCP y UDP, pero la biblioteca abstrae estos medios de comunicación, por lo que el programador solamente tiene que indicar el puerto al que desea conectarse. Los sockets son usados para establecer un canal de comunicación entre dos medios, en donde se puede enviar o recibir información.

Otro punto importante es la concurrencia de los clientes en el servidor, ya que pueden existir múltiples intentos de comunicación de clientes a un servidor en un rango de tiempo pequeño, por ejemplo, varios clientes pueden intentar comunicarse con el servidor en un mismo segundo. Para que el servidor siempre esté disponible, cada petición que hace un cliente genera un hilo en el servidor. Por lo tanto, el servidor siempre estará escuchando peticiones de clientes, y cuando llegue una nueva generará un nuevo hilo para atenderla.

4.4.2. Inicio de servidor

El servidor es el encargado de coordinar la comunicación entre los clientes que están trabajando juntos en un mismo mapa mental. Siempre existirá un servidor cuando varios clientes se encuentran colaborando en un mismo mapa mental.

También es importante dar a conocer que todos los clientes deben estar en la misma LAN, esto incluirá al servidor, ya que como se dijo éste será uno de los clientes.

Para iniciar el servidor, el usuario puede decidir activar su dispositivo como tal. En la interfaz de usuario, existe un menú dentro del área de trabajo para activar opciones globales, una de estas es la de iniciar el cliente como servidor.

Al iniciar el servidor se ejecuta el pseudocódigo del Algoritmo 3. Primero se obtiene la dirección IP del dispositivo que ha solicitado convertirse en servidor. Usando bibliotecas que acompañan a Android se puede lograr, ya que la API tiene un método que obtiene la IP.

Algorithm 3 Pseudocódigo para iniciar servidor

```
function iniciarServidor
    Obtener IP
    Iniciar Servidor con IP y puerto TCP
    Enviar IP y puerto TCP por broadcast
    Esperar peticiones de clientes
end function
```

Después, usando la biblioteca Kryonet, se inicia el servidor pasando la dirección IP y puerto TCP en el cuál se requerirá iniciar. Junto a este paso, el mismo servidor hará un broadcast a toda la LAN, dando a conocer que el servidor está iniciado en la IP y puerto TCP dado. Entonces el servidor queda activo, esperando peticiones del cliente.

En este momento, el dispositivo que ha iniciado como servidor, será aquel que compartirá su área de trabajo. Es decir, si el usuario ya estaba trabajando en el mapa mental y después inicia una sesión colaborativa con otro usuario, el trabajo del dispositivo que se inicio como servidor, será el que se va a compartir con los demás.

En las siguientes secciones se explicará el proceso que hace que un cliente entre a la sesión colaborativa con un dispositivo que ya se encuentra iniciado como servidor.

4.4.3. Conexión de clientes

Todos los dispositivos funcionan como clientes de la aplicación de mapas mentales. Cada uno de ellos puede funcionar de manera individual, pero para poder conectarse con otros dispositivos para iniciar una sesión colaborativa, debe de estar conectado a la misma LAN que los demás dispositivos con los que desea conectarse.

Algorithm 4 Conexión de un cliente a un servidor

```
function iniciarCliente
    Descubrir IP del servidor y puerto TCP
    if Cliente se conecta a servidor then
        connectToCollaborate()              ▷ Método de Algoritmo 1
    end if
end function
```

El pseudocódigo del algoritmo 4 muestra como un dispositivo se puede conectar con un vecino para llevar a cabo la remodelación del área de trabajo.

Este procedimiento hace uso de los Algoritmos 1 y 2, los cuales se discutieron en el

capítulo 3. Lo que se agregó fue que cada cliente debe de descubrir la dirección IP y el puerto TCP del servidor. Esta información se descubré por medio de la API de Kryonet, que busca servidores que estén iniciados.

Al obtener esta información, el cliente intenta conectarse con el servidor, y sí se logra, entonces el cliente procede con el Algoritmo 1, que es el que inicia el proceso de remodelación de la interfaz gráfica de usuario.

4.4.4. Paso de mensajes en una sesión colaborativa

Una vez establecida la conexión entre diversos clientes, la comunicación es coordinada por el dispositivo servidor. Cuando un usuario agrega un nodo al mapa mental o modifica algún nodo existente, el pseudocódigo del Algoritmo 5 es ejecutado. Este pseudocódigo logra la sincronización entre los distintos clientes que se encuentran conectados en la misma sesión colaborativa.

Algorithm 5 Notificación de modificación del mapa mental

function MODIFICARMAPAMENTAL
Se agrega un nodo nuevo o se modifica un nodo existente en el mapa mental
El dispositivo notifica al servidor esta acción
El servidor recibe la notificación y revisa quién la envió
El servidor envía la notificación a todos los demás clientes que están conectados en la misma sesión
Cada cliente obtiene la notificación y actualiza su mapa mental
end function

Siempre que un usuario haga algo en el mapa mental, avisa al servidor de la acción que realizó. El servidor reconoce la acción y envía la modificación del mapa mental a cada uno de los demás clientes. Entonces cada cliente deberá actualizar su mapa mental, para que vea que tenga el mismo estado del mapa mental que los demás clientes.

En el capítulo 3 se mencionó que existen dos formas para intercambiar información entre los dispositivos en una sesión colaborativa. Estas formas involucran el uso de gestos por parte de los usuarios de los dispositivos. Los gestos son *drag* y *fling*.

Un usuario envía nodos del mapa mental por medio de los gestos anteriormente mencionados. Para enviar un nodo por medio de *drag*, se usa el pseudocódigo del Algoritmo 6.

Enviar un nodo arrastrandolo funciona porque ya se ha visto que cada cliente está sincronizado con los demás, gracias al Algoritmo 5. Entonces al mover un nodo muy cerca

Algorithm 6 Enviar nodo por medio de drag

function DRAGNODO
El usuario pone su dedo sobre un nodo del mapa mental
El usuario arrastra el nodo
El sistema reconoce que se está realizando un gesto *drag*
El usuario arrastra el nodo saliendo de la pantalla y hacia el dispositivo vecino
El nodo desaparece en el dispositivo origen y aparece en el dispositivo destino
end function

de la orilla de la pantalla, hará que este nodo sea visto en el dispositivo vecino. Entonces cualquier usuario podrá arrastrar el nodo en el otro dispositivo, ya que se logra visualizar.

En el caso del gesto *fling*, se usa el pseudocódigo del Algoritmo 7. En este caso, se envía al servidor el nodo que se desea pasar al vecino. El servidor reconocerá que se está haciendo un gesto *fling* y podrá obtener el nodo que se está intentado pasar.

Algorithm 7 Enviar nodo por medio de fling

function DRAGNODO
El usuario pone su dedo sobre un nodo del mapa mental
El usuario lanza el nodo hacia el dispositivo destino
El sistema reconoce que se está realizando un gesto *fling*
El sistema reconoce la dirección del fling
if Se reconoce que en esa dirección existe un dispositivo vecino **then**
El nodo desaparece en el dispositivo origen y aparece en el dispositivo destino
end if
end function

Sí existe un dispositivo vecino, conectado en la sesión colaborativa, en la dirección en la que se realizó el gesto, entonces el nodo se moverá lógicamente de posición haciendo que desaparezca en el dispositivo origen y aparezca en el dispositivo destino.

Capítulo 5

Conclusiones y trabajo a futuro

En el último capítulo se describe a grandes rasgos lo que se realizó en este trabajo de tesis. En la sección 5.1 se explica porque se hizó la aplicación de mapas mentales y en que áreas se aporta, así como también el por qué se eligió la plataforma Android. En la sección 5.2 se muestran algunos puntos que pueden servir en futuras mejoras a la aplicación, o en trabajos que sean parecidos, es decir, que también hagan desarrollen la transición del trabajo individual al colaborativo.

5.1. Conclusiones

Actualmente existen pocos trabajos que hayan abordado la plasticidad de las interfaces gráficas de usuario tomando en cuenta sus dos medios de adaptación. El sistema de mapas mentales hace uso de la remodelación plástica y la redistribución plástica, por lo que toma en cuenta variables de contexto como la plataforma en la que se está ejecutando y el usuario.

Este trabajo también hace un aporte en las aplicaciones que funcionan indivdualmente, pero que el usuario puede decidir trabajar colaborativamente con otra persona que se encuentre en el mismo lugar. Se intenta que la transición del trabajo individual a colaborativo y viceversa, sea lo más fluida posible para que sea lo más natural para el usuario y no obstruya con lo que esté haciendo el usuario.

Por lo tanto, este trabajo tiene en cuenta que el punto central en las aplicaciones colaborativas debe de ser el usuario, porque es importante la retroalimentación que el sistema le debe brindar al usuario, ya que nunca debe obstruir con sus actividades. A fin de cuentas, las aplicaciones son para ayudar con las actividades del usuario.

Se decidió trabajar sobre la plataforma Android debido a que es gratuita y hecho muy popular en los últimos años, y se intenta que la aplicación de mapas mentales pueda ser usada por la mayor cantidad de personas.

Se analizó y diseño la arquitectura más factible para el desarrollo de la aplicación de mapas mentales. Se siguió el patrón arquitectural MVC, el cuál es fácil de implementar en las plataformas móviles, en específico Android, ya que una de sus virtudes es el uso de un lenguaje de programación orientado a objetos, en este caso en Android se usa Java.

Se probó la aplicación tanto con tabletas y smartphones Android, de diferentes tamaños de pantalla para comprobar el correcto funcionamiento de las interfaces de usuario en dispositivos con características diferentes.

Las interfaces de usuario fueron prediseñadas por el programador, así que en tiempo de ejecución se debe de decidir que interfaz mostrar dependiendo de las características del dispositivo en el que se este ejecutando la aplicación. Sin embargo, cuando se hace la redistribución de la interfaz de usuario entre varios dispositivos, también se hace una remodelación por cada dispositivo que está conectado en una sesión colaborativa.

El área de dibujo es el componente que se redistribuye entre dos o más dispositivos, y se puede redistribuir entre smartphones y tablets. El tamaño del área de trabajo en cada dispositivo, dependerá del tamaño de su pantalla, aunque podrá desplazarse dentro de esta área por medio de gestos.

La aplicación de mapas mentales es usada por medio de gestos, ya que casi todos los dispositivos modernos cuentan con pantalla táctil. Existen diversos gestos en la aplicación que llevan a cabo diferentes acciones, como la manipulación del árbol mental, y el envío de elementos entre los dispositivos que están conectados en una sesión colaborativa.

La aplicación se comunica usando una topología cliente-servidor, por lo que en una sesión colaborativa siempre existe un servidor, que al mismo tiempo es uno de los clientes que están dentro de la colaboración.

5.2. Trabajo a futuro

Se encontró que la aplicación de mapas mentales podría mejorar en los siguientes aspectos:

- Implementar otro método de entrada diferente al tacto, ya que algunos dispo-

sitivos podrían no tener una pantalla táctil.

- Crear un módulo que permita a la aplicación funcionar también de manera remota, para que los colaboradores también puedan trabajar en diferentes lugares.
- Explorar e implementar diferentes mecanismos para detectar dispositivos cercanos e iniciar una sesión colaborativa.
- Permitir al usuario dibujar a mano alzada para que tenga más libertad en sus gráficos desarrollados con la aplicación.
- Exportar los gráficos creados con la aplicación a imágenes o algún archivo como pdf.
- Los dispositivos podrán comunicarse usando peer to peer, por lo que no será necesario un servidor.
- Implementar la aplicación en más plataformas, como iOS.

También se identificó que sería bueno contar con un framework, que pueda ser usado para realizar aplicaciones que necesiten hacer la transición de un estado individual a un estado colaborativo. Este framework debería contar con eventos que, por ejemplo, indiquen al programador cuando exista un dispositivo cercano con el que se pueda conectar.

Bibliografía

[Bailey et al., 2008] Bailey, B. P., Biehl, J. T., Cook, D. J., and Metcalf, H. E. (2008). Adapting paper prototyping for designing user interfaces for multiple display environments. *Personal and Ubiquitous Computing*, 12(3):269–277.

[Baker et al., 2002] Baker, K., Greenberg, S., and Gutwin, C. (2002). Empirical Development of a Heuristic Evaluation Methodology for Shared Workspace Groupware. In *Proceedings of the 2002 ACM conference on Computer supported cooperative work*, pages 96–105. ACM.

[Browne et al., 1990] Browne, D., Totterdell, P., and Norman, M. (1990). *Adaptative user interfaces*. Academic Press Ltd. London, UK.

[Calvary et al., 2006] Calvary, G., Coutaz, J., Daasi, O., Ganneau, V., Balme, L., Demeure, A., and Sottet, J.-S. (2006). Métamorphose des IHM et Plasticité: Article de synthèse. In *Ergo'IA 2006*, pages 11–18.

[Calvary et al., 2001] Calvary, G., Coutaz, J., and Thevenin, D. (2001). A Unifying Reference Framework for the Development of Plastic User Interfaces. In *Proceedings of the 8th IFIP International Conference on Engineering for Human-Computer Interaction*, pages 173–192. Springer-Verlag London.

[Coutaz, 2010] Coutaz, J. (2010). User Interface Plasticity: Model Driven Engineering to the Limit! In *Proceedings of the 2nd ACM SIGCHI symposium on Engineering interactive computing systems*, pages 1–8. ACM Press.

[Coutaz and Calvary, 2012] Coutaz, J. and Calvary, G. (2012). HCI and Software Engineering for User Interface Plasticity. In *Human-Computer Interaction Handbook: Fundamentals, Evolving Technologies, and Emerging Applications*, pages 1195–1220. CRC Press Taylor and Francais Group.

[Demeure et al., 2005] Demeure, A., Calvary, G., Sottet, J.-S., and Vanderdonkt, J. (2005). A Reference Model for Distributed User Interfaces. In *Proceedings of the*

4th international workshop on Task models and diagrams, pages 79–86. ACM.

[Dey, 2001] Dey, A. K. (2001). Understanding and Using Context. *Personal and Ubiquitous Computing*, 5(1):4–7.

[Ellis et al., 1991] Ellis, C. A., Gibbs, S. J., and Rein, G. (1991). Groupware: some issues and experiences. *Communications of the ACM*, 34(1):39–58.

[Greif, 1988] Greif, I. (1988). *Computer-supported Cooperative Work*. Morgan Kaufmann.

[Haake et al., 2010] Haake, J. M., Hussein, T., Joop, B., Lukosch, S., Veiel, D., and Ziegler, J. (2010). Modeling and Exploiting Context for Adaptive Collaboration. *International Journal of Cooperative Information Systems*, 19(1-2):71–120.

[Herskovic et al., 2008] Herskovic, V., Ochoa, S. F., Pino, J. A., and Neyem, A. (2008). General Requirements to Design Mobile Shared Workspaces. In *12th International Conference on Computer Supported Cooperative Work in Design, 2008. CSCWD 2008.*, pages 582–587.

[Hinckley, 2003] Hinckley, K. (2003). Synchronous Gestures for Multiple Persons and Computers. In *Proceedings of the 16th annual ACM symposium on User interface software and technology*, pages 149–158. ACM.

[Hinckley et al., 2004] Hinckley, K., Ramos, G., Guimbretiere, F., Baudisch, P., and Smith, M. (2004). Stitching: Pen Gestures that Span Multiple Displays. In *Proceedings of the working conference on Advanced visual interfaces*, pages 23–31. ACM.

[Hollan et al., 2000] Hollan, J., Hutchins, E., and Kirsh, D. (2000). Distributed Cognition: Toward a New Foundation for Human-Computer Interaction Research. *ACM Transactions on Computer-Human Interaction*, 7(2):174–196.

[Johanson et al., 2002] Johanson, B., Fox, A., and Winograd, T. (2002). The Interactive Workspaces Project: Experiences with Ubiquitous Computing Rooms. *IEEE Pervasive Computing*, 1(2):67–74.

[Ketabdar et al., 2010] Ketabdar, H., Yüksel, K. A., and Roshandel, M. (2010). MagiTact: Interaction with Mobile Devices Based on Compass (Magnetic) Sensor. In *Proceedings of the 15th international conference on Intelligent user interfaces*, pages 413–414. ACM.

[Kuechler and Kunz, 2010] Kuechler, M. and Kunz, A. M. (2010). CollaBoard: A Remote Collaboration Groupware Device Featuring an Embodiment-enriched Shared

Workspace. In *Proceedings of the 16th ACM International conference on Supporting group work*, pages 211–214. ACM Press.

[Li and Kobbelt, 2012] Li, M. and Kobbelt, L. (2012). Dynamic Tiling Display: Building an Interactive Display Surface using Multiple Mobile Devices. In *11th International Conference on Mobile and Ubiquitous Multimedia*.

[Lucero et al., 2010] Lucero, A., Keränen, J., and Korhonen, H. (2010). Collaborative Use of Mobile Phones for Brainstorming. In *Proceedings of the 12th international conference on Human computer interaction with mobile devices and services*, pages 337–340. ACM.

[Luyten et al., 2007] Luyten, K., Verpoorten, K., and Coninx, K. (2007). Ad-hoc Co-located Collaborative Work with Mobile Devices. In *Proceedings of the 9th international conference on Human computer interaction with mobile devices and services*, pages 507–514. ACM.

[Madhavapeddy and Tse, 2005] Madhavapeddy, A. and Tse, A. (2005). A Study of Bluetooth Propagation Using Accurate Indoor Location Mapping. In *Proceedings of the 7th international conference on Ubiquitous Computing*, pages 105–122. Springer-Verlag Heidelberg.

[Morales, 2009] Morales, G. S. (2009). Redistribución semi-plástica mixta: el caso de estudio de un pizarrón compartido. Tesis de maestría, Centro de Investigación y de Estudios Avanzados del IPN, Departamento de Computación.

[Prante et al., 2004] Prante, T., Streitz, N. A., and Tandler, P. (2004). Roomware: Computers Disappear and Interaction Evolves. *Computer*, 37(12):47–54.

[Rekimoto, 1997] Rekimoto, J. (1997). Pick-and-Drop: A Direct Manipulation Technique for Multiple Computer Environments. In *Proceedings of the 10th annual ACM symposium on User interface software and technology*, pages 31–39. ACM.

[Rekimoto and Saitoh, 1999] Rekimoto, J. and Saitoh, M. (1999). Augmented Surfaces: A Spatially Continous Work Space for Hybrid Computing Environments. In *Proceedings of the ACM CHI 99 Human Factors in Computing Systems Conference*, pages 378–285. ACM Press.

[Schilit et al., 1994] Schilit, B. N., Adams, N., and Want, R. (1994). Context-Aware Computing Applications. In *Proceedings of the 1994 First Workshop on Mobile Computing Systems and Applications*, pages 85–90. IEEE Computer Society.

[Schmidt et al., 2012] Schmidt, A., Pfleging, B., Alt, F., Shirazi, A. S., and Fitzpa-

trick, G. (2012). Interacting with 21st-Century Computers. *Pervasive Computing*, 11(1):22–31.

[Sottet et al., 2006] Sottet, J.-S., Calvary, G., Favre, J.-M., Coutaz, J., Demeure, A., and Balme, L. (2006). Towards Model Driven Engineering of Plastic User Interfaces. In *Satellite Events at the MoDELS 2005 Conference*, pages 191–200. Springer Berlin Heidelberg.

[Streitz et al., 2002] Streitz, N., Prante, T., Müller-Tomfelde, C., Tandler, P., and Magerkurth, C. (2002). Roomware - The Second Generation. In *Proceedings of of CHI 2002 on Extended Abstracts on Human Factors in Computing Systems*, pages 506–507. ACM.

[Tandler et al., 2001] Tandler, P., Prante, T., Müller-Tomfelde, C., Streitz, N., and Steinmetz, R. (2001). ConnecTables: Dynamic Coupling of Displays for the Flexible Creation of Shared Workspaces. In *Proceedings of the 14. Annual ACM Symposium on User Interface Software and Technology, (UIST '01)*, pages 11–20, New York, NY, USA. ACM.

[Terrenghi et al., 2009] Terrenghi, L., Quigley, A., and Dix, A. (2009). A taxonomy for and analysis of multi-person-display ecosystems. *Personal and Ubiquitous Computing*, 13(8):583–598.

[Thevenin and Coutaz, 1999] Thevenin, D. and Coutaz, J. (1999). Plasticity of User Interfaces: Framework and Research Agenda. In *Proceedings of INTERACT 99 - IFIP TC13 Seventh International Conference on Human-Computer Interaction*, pages 1–10. IOS Press.

[Vanderdonckt and Calleros, 2008] Vanderdonckt, J. and Calleros, J. M. G. (2008). Task-Driven Plasticity: One Step Forward with UbiDraw. In *Engineering Interactive Systems. Second Conference on Human-Centered Software Engineering, HCSE 2008, and 7th International Workshop on Task Models and Diagrams, TAMODIA 2008*, pages 181–196, Pisa, Italia. Springer.

[W. Greg Phillips, 1999] W. Greg Phillips (1999). Architectures for Synchronous Groupware. Technical report, .

[Weiser, 1991] Weiser, M. (1991). The Computer for the 21st Century. *Scientific American*, 265(3):94–104.

[Yuill and Rogers, 2012] Yuill, N. and Rogers, Y. (2012). Mechanisms for Collaboration: A Design and Evaluation Framework for Multi-User Interfaces. *ACM Transactions on Computer-Human Interaction*, 19(1):1–25.

[Zimmermann et al., 2007] Zimmermann, A., Lorenz, A., and Oppermann, R. (2007). An Operational Definition of Context. In *6th International and Interdisciplinary Conference, CONTEXT 2007*, pages 558–571. Spring-Verlag Heidelberg.