CENTRO DE INVESTIGACIÓN Y DE ESTUDIOS AVANZADOS
DEL INSTITUTO POLITÉCNICO NACIONAL

Unidad Zacatenco

Departamento de Computación

"Arquitectura para el soporte de contexto de uso en sistemas colaborativos"

Tesis que presenta

Anallely Olivares Toledo

Para Obtener el Grado de

Maestra en Ciencias

en Computación

Directores de Tesis

Dr. Sonia Guadalupe Mendoza Chapa

Dr. Adriano de Luca Pennacchia

México, Distrito Federal Diciembre, 2011

Resumen

Los entornos de trabajo colaborativo se caracterizan por las diferentes situaciones que pueden ocurrir en distintos momentos. Estas situaciones son el producto de la interacción entre varias personas, quienes poseen habilidades diversas y en consecuencia juegan diferentes roles en proyectos comunes. A su vez, estos proyectos están regidos por fechas de entrega bien definidas y requieren recursos de disponibilidad variable que utilizan los colaboradores de manera oportunista durante la realización de cada tarea. El software que da soporte a la colaboración (*groupware*) debería ser lo suficientemente flexible para adaptarse a cada situación, con la intención de incrementar su usabilidad y capacidades para impactar positivamente el desempeño general del grupo.

Para ilustrar este requerimiento, considere que varias personas trabajan en el mismo proyecto. En este caso, el sistema colaborativo debería estar consciente de las necesidades de cada usuario para adaptarse convenientemente, e.g., destacando los documentos que le son más útiles a cada uno. Otra situación común en entornos de trabajo colaborativo ocurre cuando la fecha de entrega de un proyecto está próxima; en este caso el sistema colaborativo debería adaptarse para dar prioridad al proyecto en cuestión, e.g., facilitando herramientas para agilizar su desarrollo y eliminando los elementos que causan distracciones. Un ejemplo más se observa cuando se está llevando a cabo una reunión importante; en esta situación el sistema colaborativo podría adaptarse para no admitir interrupciones, e.g., posponiendo la entrega de mensajes irrelevantes a un momento más apropiado.

El requerimiento de adaptación se relaciona con el concepto de "contexto de uso", el cual es un elemento importante en el diseño de sistemas interactivos. Sin embargo, dos cuestiones relativas a este concepto han sido identificadas en las investigaciones actuales sobre cómputo consciente de contexto: 1) la mayoría de los estudios se han centrado en el contexto de un solo usuario, así que el contexto de múltiples usuarios involucrados en un mismo proyecto permanece prácticamente inexplorado y 2) la adaptabilidad de los sistemas contextuales considera un número reducido de variables (principalmente la ubicación del usuario y la plataforma). La presente tesis se centra en estudiar el contexto de uso en entornos de trabajo colaborativo, enfatizando la importancia de la adaptabilidad de los sistemas colaborativos con base en diversas variables típicas de los grupos de trabajo, tales como el estado de los proyectos, políticas organizacionales, la ubicación física de los colaboradores y recursos disponibles. Para soportar e integrar el contexto de uso en sistemas colaborativos, se propone una arquitectura contextual basada en escenarios y situaciones reales que sirven como medio para validar su funcionalidad.

Palabras clave: adaptabilidad, contexto de uso, cómputo consciente de contexto, entornos de trabajo colaborativo, sistemas colaborativos.

Abstract

Collaborative working environments are affected by different situations occurring at any moment. These situations come from the interaction among several persons who have different skills and thus play different roles in common projects. In turn, these projects are subject to well defined delivery dates and require variable available resources used by collaborators on an opportunistic manner in the procedures needed to accomplish each task. The software for supporting collaborative work (groupware) should be flexible enough to adapt itself to each situation, with the aim of increasing its usability and capabilities to positively impact the overall performance of the group.

In order to illustrate this requirement, let us consider that several persons work on the same project. In this case, the groupware system should be aware of each user's needs to adapt itself accordingly, e.g., by highlighting the documents most useful for each one. Another common situation in colaborative working environments occurs when the delivery date of a project is nearby; in this case, groupware system should adapt itself to give a greater priority to the project in question, e.g., by providing tools that speed up its development and by eliminating distracting elements. Another example is observed when an important meeting is taking place; in this situation the groupware system should adapt itself to not admit interruptions, e.g., by delaying the delivery of irrelevant messages to a more appropriate time.

This requirement is related to the concept of "context of use", which is an important element in the design of interactive systems. However, two issues about this concept have been identified in current researches on context-aware computing: 1) most of the studies have mainly focused on the context of a single user, so the context of multiple users involved in a common project practically remains unexplored, and 2) the adaptability in context-aware systems generally considers a reduced number of variables (mainly user's location and platform). The present thesis is focused on studying the context of use in collaborative working environments, emphasizing the importance of groupware system adaptability depending on several typical variables of the working groups, such as the state of projects, organizational policies, the collaborators' physical location, and available resources. In order to support and integrate the context of use in groupware systems, we propose a contextual architecture based on real scenarios and situations that serve as a means of validating its functionality.

Key words: adaptability, collaborative working environments, context-aware computing, context of use, groupware systems.

Agradecimientos

Al CONACyT (Consejo Nacional de Ciencia y Tecnología)

Ya que sin la ayuda del apoyo económico proporcionado por este organo, esta tesis de maestría no habría podido completarse.

Al CINVESTAV

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

A la Dra. Sonia

Por aceptar asesorarme en este trabajo de tesis. En especial, por su dedicación y paciencia durante las revisiones de cada capítulo.

Al Dr. Adriano de Luca

Por aceptar ser mi co-director de tesis y brindar su apoyo incondicional.

A mis padres:

Por que no importa lo que suceda siempre han estado conmigo, apoyándome en cada paso, abrazándome y escuchándome. Realmente los quiero mucho y valoraré sus enseñanzas y sacrificios cada día de mi vida.

A mi hermanito:

Por siempre preocuparse por mi bienestar y superación, además de ser una persona muy valiosa.

A mis amigos del cinves:

Con ustedes pasé ratos muy agradables: reímos, comimos, sufrimos y hasta nos desvelamos juntos, casi siempre trabajando... Conocer personas como ustedes fue una de las mejores experiencias en mi vida y espero no perderlos de vista.

A todos los profesores del departamento de computación:

Fue totalmente gratificante cada reto, cada tarea, cada enseñanza brindada tanto en los cursos, como fuera de ellos.

A todo el personal administrativo del departamento de computación:

En especial a Sofy Reza, por su paciencia infinita y su amabilidad en cada paso dado através de nuestra estancia en el programa de maestría.

A Dios:

Dios, tu sabes que siempre te tengo presente. Gracias de nuevo.

Índice general

Índice de figuras

Índice de cuadros

Capítulo 1

Introducción

Nuestro comportamiento se adapta a las distintas situaciones que se presentan en la vida diaria, e.g., nos apresuramos si estamos retrasados para una reunión, levantamos nuestra voz para ser escuchados en ambientes ruidosos y evitamos distractores cuando tenemos una gran cantidad de trabajo pendiente. Por el contrario, la mayoría de los sistemas computacionales no se adaptan al contexto del usuario, o bien, lo hacen de una mánera muy básica, e.g., algunas características de los teléfonos celulares pueden ser modificadas dependiendo del perfil activo del usuario.

La capacidad de adaptación de los sistemas se relaciona con el concepto de "contexto de uso", el cual ha sido considerado como un elemento importante en los sistemas computacionales desde mediados de la década de los ochenta [Coutaz and Rey, 2002]. Sin embargo, este concepto ha ganado mayor importancia en años recientes gracias a la creciente aceptación y accesibilidad de dispositivos móviles (e.g., teléfonos celulares, laptops y tabletas electrónicas) y a su importante rol en el área del Cómputo Ubicuo en donde la información contextual (e.g., hora, ubicación, usuarios y recursos disponibles) es usada para representar aspectos del mundo físico [Haake et al., 2010].

El presente trabajo de tesis aborda el concepto de contexto de uso desde el punto de vista de los entornos colaborativos, en los cuales varios usuarios interactúan con la ayuda de un sistema computacional que coordina y sincroniza sus actividades para alcanzar un objetivo en común. Los entornos colaborativos son altamente dinámicos y solo recientemente han sido tomados en cuenta en las investigaciones sobre el contexto de uso [Brézillon et al., 2008].

Específicamente, en este documento se propone una arquitectura contextual para soportar la detección de cambios en el contexto de uso, con el objetivo de que los sistemas computacionales se adapten a ellos. Dicha arquitectura está dirigida particularmente a sistemas colaborativos (*groupware*), i.e., sistemas que soportan la colaboración de múltiples usuarios. Por lo tanto, se toman en cuenta variables contextuales como el estado de los proyectos, las políticas organizacionales, la ubicación física de los colaboradores, los recursos disponibles y otras variables típicas en los grupos de trabajo.

La arquitectura contextual que se propone está basada en escenarios y situaciones comunes en los entornos colaborativos, que a la vez sirven como medio para validar su funcionalidad.

1.1 Contexto de investigación

La presente tesis de maestría se inscribe en el dominio de investigación del área de Trabajo Colaborativo Asistido por Computadora (TCAC o CSCW[1] por sus siglas en inglés), la cual se centra en resolver problemas relativos a la forma en que los miembros de un equipo de trabajo se comunican, se coordinan y colaboran para alcanzar un objetivo en común.

TCAC es un área multi-disciplinaria que estudia tanto los aspectos sociales de las actividades individuales y colectivas, como los aspectos tecnológicos que soportan la colaboración entre personas. El presente trabajo de investigación se enfoca principalmente en la perspectiva tecnológica del diseño y de la implementación de una arquitectura para facilitar el desarrollo de sistemas colaborativos capaces de adaptarse al contexto en el que se ejecutan (cf. figura 1.1).

Figura 1.1: Contexto de investigación de la presente tesis

Un **sistema colaborativo**, también conocido como software colaborativo o *groupware*, proporciona los medios y herramientas que permiten que múltiples usuarios trabajen sobre el mismo proyecto de manera simultánea [Rama and Bishop, 2006]. Los sistemas colaborativos son importantes debido a que la mayoría de las actividades que se realizan en las organizaciones no son efectuadas por una sola persona, sino que detrás de cada tarea se encuentra un equipo de colaboradores quienes trabajan conjuntamente para poder alcanzar un objetivo en común. El desarrollo tecnológico en materia de dispositivos móviles, así como el progreso de las redes de comunicación han

[1] Computer-Supported Cooperative Work

permitido que la colaboración sea posible aún si los participantes no se encuentran en el mismo lugar físico y/o al mismo tiempo.

Un **entorno colaborativo** se refiere al espacio físico y lógico en donde se llevan a cabo las actividades de colaboración de un grupo de trabajo. Los entornos colaborativos son dinámicos: es posible que no todos los colaboradores estén co-localizados temporal o espacialmente, además las asignaciones de responsabilidades y los roles están sujetos a cambios. Los usuarios son entes que evolucionan en un entorno variable y que utilizan, de manera oportunista, plataformas de interacción diversas con el fin de satisfacer sus necesidades en cualquier lugar donde se encuentren [Calvary et al., 2001].

Un entorno dinámico como el que se describe impone un nuevo requerimiento: la capacidad de adaptación del software que da soporte a la colaboración a las distintas situaciones que se presentan. El requerimiento anterior se relaciona con el concepto de **contexto de uso**. A pesar de que muchos artículos en la literatura científica hacen referencia al contexto no existe una definición clara y ampliamente aceptada por toda la comunidad. Sin embargo, Day propone la siguiente definición general, la cuál es probablemente la más difundida [Abowd et al., 1999]:

> "Contexto es cualquier información que puede ser usada para caracterizar la situación de una entidad. Una entidad es una persona, lugar u objeto que es considerada relevante para la interacción entre un usuario y una aplicación, incluyendo al usuario y a la aplicación misma."

1.2 Planteamiento del problema

En la literatura científica se reporta mucho trabajo enfocado a definir, modelar y percibir cambios en el contexto de uso en un sistema computacional. Sin embargo, la mayor parte de la investigación se ha llevado a cabo desde el punto de vista de los sistemas mono-usuario, dando un mayor énfasis a variables contextuales como la plataforma (más específicamente el tamaño de la pantalla de despliegue) y la ubicación física de los usuarios. FlexClock [Grolaux et al., 2002] y UbiCicero [Ghiani et al., 2009] son ejemplos de este tipo de sistemas.

En 2008, Brézillon et al. advierten que la investigación sobre el contexto raramente considera aspectos prácticos de las actividades de la vida real como el trabajo colaborativo [Brézillon et al., 2008]. Destacan que casi todos los esfuerzos se han centrado en elementos físicos del contexto de uso como la hora, el clima y otro tipo de información que puede ser obtenida a través de sensores y usada directamente en las aplicaciones. Más aún, concluyen que la mayoría de los estudios generalmente consideran el contexto de un solo usuario, por lo que el contexto de un grupo colaborativo con múltiples usuarios sigue siendo un tema prácticamente inexplorado.

En los entornos colaborativos participan personas con diferentes antecedentes y habilidades, que usan diferentes mecanismos de comunicación, coordinación y pro-

ducción dependiendo de los lugares en los que se encuentran, las actividades que deben llevar a cabo y los proyectos en los que trabajan. Además de estos factores existen otros, como los roles de usuario, que hacen que los entornos colaborativos sean altamente dinámicos.

El presente documento propone una arquitectura derivada de la investigación del contexto desde el punto de vista de entornos colaborativos, dando un mayor énfasis en las variables que son propias de grupos (e.g., los roles de trabajo, el estado de los proyectos, las políticas y los procedimientos organizacionales, etc.).

1.3 Motivación y objetivos de la tesis

Con el presente trabajo se pretende responder a las siguientes preguntas: ¿Se obtienen ventajas al considerar el contexto de uso en sistemas colaborativos? ¿Cómo modelar adecuadamente los cambios contextuales en este tipo de sistemas? Concluímos que la respuesta a la primera pregunta es afirmativa, como lo muestra el siguiente escenario:

"Suponga que un grupo de colaboradores trabaja en el diseño de un logotipo para un nuevo producto que un cliente está a punto de lanzar al mercado. Todos se encuentran reunidos en una sala de juntas. En un momento dado, una persona entra a la oficina. El sistema colaborativo que están usando para alcanzar su objetivo detecta que la persona que acaba de entrar es externa al equipo de colaboración, por lo que se encarga de ocultar (o alterar) la información sensible, e.g., nombre del producto, imagen y logotipo. Cuando la persona externa sale de la sala de juntas, la información se despliega de nuevo".

En el escenario descrito se observa que el contexto aumenta las capacidades del sistema en forma útil. De este modo, los usuarios ya no tienen la tarea de ocultar por sí mismos la información a la persona externa, sino que esta acción se realiza automáticamente.

Objetivo general

Definir una arquitectura para facilitar la integración del contexto de uso en sistemas colaborativos; dicha arquitectura debe ser capaz: a) de facilitar el desarrollo de sistemas colaborativos sensibles al contexto y b) de soportar cambios tanto en variables físicas como lógicas en los entornos colaborativos.

Objetivos particulares

1. Proponer escenarios y situaciones en donde se muestre cómo, al explotar el contexto de uso, se puede incrementar las capacidades y la usabilidad de los sistemas colaborativos.

2. Con base en dichos escenarios, desarrollar una arquitectura para modelar el contexto de uso en sistemas colaborativos, que además soporte la detección de cambios en el contexto de uso, con el objetivo de que los sistemas computacionales los consideren y se adapten a ellos.

3. Desarrollar un sistema colaborativo sensible al contexto que valide la funcionalidad de la arquitectura propuesta.

1.4 Organización de la tesis

El mapa del presente documento de tesis se muestra en la figura 1.2. El primer capítulo corresponde a esta introducción en donde se plantea el contexto de investigación, el planteamiento del problema, la motivación y los objetivos generales y específicos de esta investigación.

El capítulo 2 comprende el estado del arte, en donde se describe desde tres perspectivas distintas los principales resultados publicados, que están relacionados con el tema de investigación. La primera parte explica las definiciones más relevantes del concepto de "contexto de uso", así como la forma en que este concepto es aplicado y modelado en sistemas computacionales. La segunda parte describe ejemplos específicos de sistemas y aplicaciones sensibles al contexto de uso. Finalmente, se mencionan las características, ventajas y limitaciones de *frameworks* relacionados con la presente tesis.

Las contribuciones del presente trabajo se reflejan en los capítulos 3 y 4.

El capítulo 3 corresponde a la etapa de análisis de casos de uso y diseño de la arquitectura contextual propuesta. El análisis consiste en la descripción de escenarios en donde el contexto de uso enriquece las funcionalidades de un sistema colaborativo. Algunos de estos escenarios se retoman de trabajos relacionados previos, pero también se incluyen propuestas que no han sido estudiadas con anterioridad. En la sección de diseño se describe la arquitectura propuesta, consistente de cuatro fases: 1) percepción de eventos contextuales, 2) detección de situaciones, 3) comunicación de situaciones y 4) adaptación del sistema colaborativo. Cada fase de la arquitectura se describe con detalle.

En el capítulo 4 se presenta una implementación basada en el diseño de la arquitectura propuesta. Esta implementación considera aspectos como el lenguaje de programación, estructuras de datos, módulos, herramientas y tecnologías, sin embargo, estos elementos pueden ser reemplazados por los más adecuados dependiendo de las necesidades específicas del sistema colaborativo sobre el que se implemente la arquitectura. En este capítulo también se describe la aplicación de prueba desarrollada, un editor colaborativo de mapas mentales.

Finalmente en el capítulo 5, se presenta las conclusiones, contribuciones y el trabajo futuro que se deriva de la presente tesis.

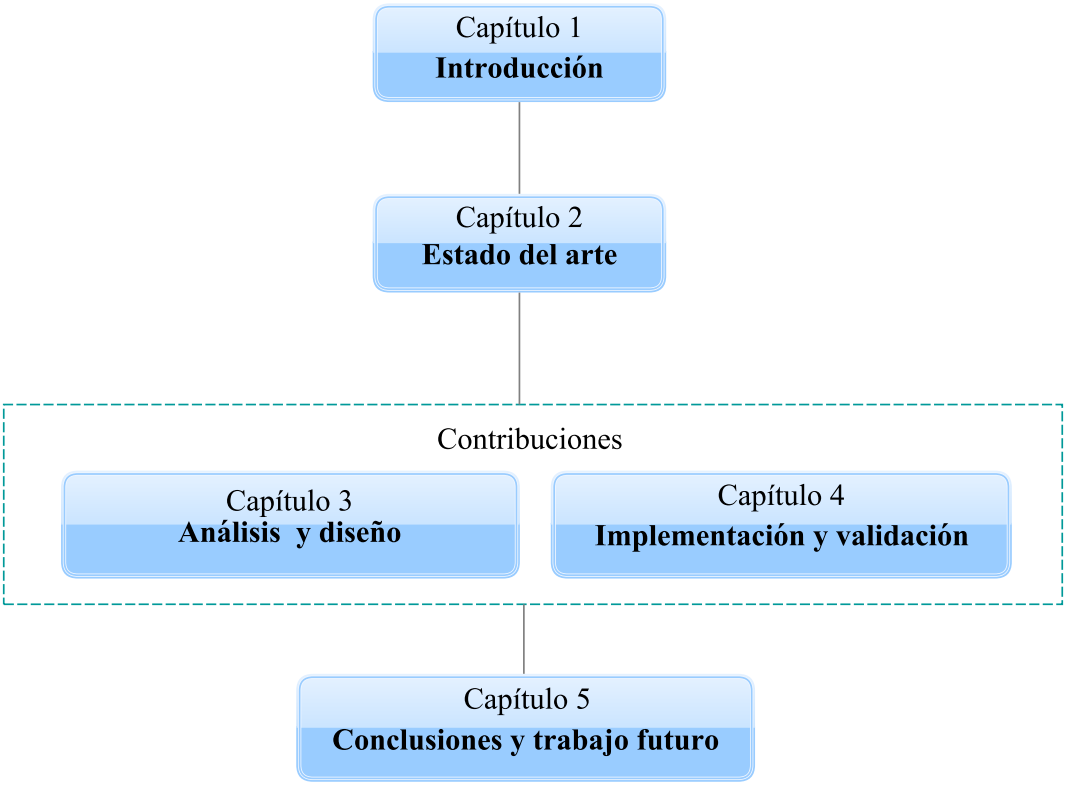


Figura 1.2: Organización del documento de tesis

Capítulo 2

Estado del arte

El presente capítulo se compone de tres partes que cubren diferentes aspectos significativos para la presente tesis. La primera parte explica las definiciones más importantes del concepto de "contexto de uso", así como los principales resultados publicados sobre la forma en que este concepto es aplicado y modelado en los sistemas computacionales (sección 2.1). La segunda parte describe ejemplos específicos de sistemas y aplicaciones sensibles al contexto de uso (sección 2.2). Finalmente en la tercera parte del capítulo se pone en evidencia las características, ventajas y limitaciones de algunos *frameworks* propuestos para el desarrollo de sistemas que se adaptan al contexto de uso (sección 2.3).

2.1 Contexto de uso

El "contexto" ha sido considerado como un concepto importante en los sistemas computacionales desde mediados de los años ochenta [Coutaz and Rey, 2002], cobrando mayor importancia en años recientes gracias a la proliferación de dispositivos móviles y redes digitales, ya que mediante el uso de estas tecnologías los sistemas dejan de estar confinados en un entorno específico, al mismo tiempo que crece la necesidad de que sean sensibles al contexto y se adapten a las circunstancias que los rodean.

El contexto juega un papel importante en áreas como el Cómputo Ubicuo, en la cual los sistemas deben considerar información contextual como la ubicación, la hora, los recursos disponibles y los usuarios para representar aspectos del mundo físico, procesando esta información para integrarla a los objetos y actividades de la vida cotidiana [Haake et al., 2010].

Una gran cantidad de artículos en la literatura científica ha hecho referencia al contexto, sin embargo no existe una definición ampliamente aceptada por todos los autores [Coutaz and Rey, 2002]. Day propone la siguiente definición general, la cual es probablemente la más difundida [Abowd et al., 1999]:

> "Contexto es cualquier información que puede ser usada para caracterizar la situación de una entidad. Una entidad es una persona, un lugar u objeto que es considerada relevante para la interacción entre un usuario y una aplicación, incluyendo al usuario y a la aplicación misma".

De acuerdo a Coutaz y Calvary, el contexto no es simplemente un estado, sino un estado que forma parte de un proceso o propósito, razón por la cual se habla de "contexto de uso" y no simplemente de "contexto" [Coutaz and Calvary, 2008]. Esta diferenciación permite tomar en cuenta solo aquellos estados (contextos) que pueden ser identificados y usados por el sistema e ignorar los demás estados.

La estrategia de algunos autores para definir "contexto de uso" ha sido clasificar y enumerar ejemplos. A manera de ilustración, Schilit et al., proponen dividir el contexto de uso en tres categorías: *contexto computacional* (e.g., conectividad, ancho de banda, impresoras y pantallas), *contexto de usuario* (e.g., perfil, ubicación y personas cercanas) y *contexto físico* (e.g., nivel de luz, temperatura y cantidad de ruido) [Schilit et al., 1994].

Una definición más formal es proporcionada por Calvary et al., quienes describen al contexto de uso como la terna <usuario, plataforma, entorno> [Calvary et al., 2004] [Coutaz and Calvary, 2008] donde:

1. el **usuario** denota el arquetipo de la persona que interactúa con el sistema. El modelo del usuario incluye su perfil, su idiosincrasia, sus tareas y sus actividades;

2. la **plataforma** se refiere a los recursos de hardware y software disponibles para sustentar la interacción del usuario con el sistema. El modelo de la plataforma puede ser descrito en términos de sensores, redes de comunicación y recursos que unen el ambiente físico con el mundo digital;

3. el **entorno** describe las condiciones físicas y sociales donde la interacción toma lugar. El modelo del entorno incluye la ubicación numérica o simbólica del usuario (e.g., en casa, en un lugar público, en movimiento en la calle), reglas sociales, actividades y condiciones de luz y sonido.

Coutaz y Rey coinciden en señalar que el contexto no puede existir de forma aislada, sino que debe ser definido con respecto a entidades particulares como el usuario, el sistema o la tarea a realizar. Ellos definen al contexto de uso como una composición de múltiples situaciones en un cierto periodo de tiempo [Coutaz and Rey, 2002]. Proponen el modelo de "contextores" para representar la relación entre las variables de un sistema contextual. El elemento básico de los contextores abstrae la información de un sensor (cf. figura 2.1) y mediante otros contextores especializados ("fusionadores", "traductores" y "disparadores") es posible crear una red que relacione todos los sensores para obtener un contexto funcional.

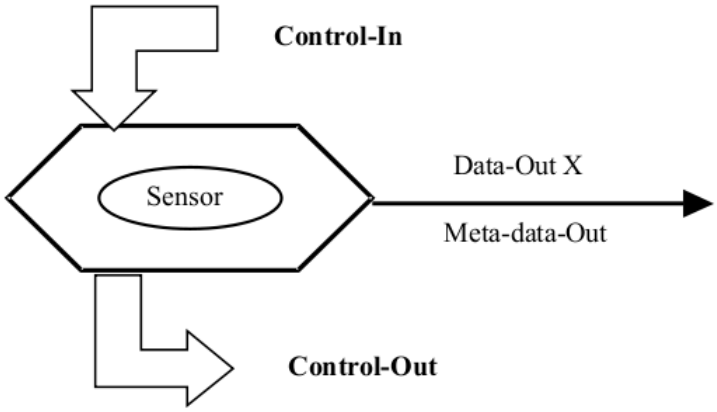


Figura 2.1: Representación gráfica del contextor elemental, cuya tarea es encapsular y abstraer la información de un sensor físico, e.g. un termómetro

De acuerdo a Crowley et al., el contexto de uso puede describirse en función de una red de roles y relaciones [Crowley et al., 2002]. Diferentes configuraciones de roles y relaciones corresponden a diferentes situaciones contextuales. Por ejemplo, el enunciado "Alicia expone un plan de negocios" describe una situación en la que los posibles roles son "expositor", "público", "superficie de despliegue", "apuntador", etc., mientras que las posibles relaciones son "señalar a", "hablar a", "escuchar a", etc. (cf. figura 2.2). La ontología que se describe en su trabajo provee una base teórica para el desarrollo de una arquitectura de software para sistemas sensibles al contexto.

Figura 2.2: En la situación de una presentación existen varios roles como "expositor" y "público", así como varias relaciones como "hablar a", "escuchar a", etc.

En 2008, Brézillon et al. advierten que la investigación sobre el contexto raramente considera aspectos prácticos de las aplicaciones de la vida real como el trabajo colaborativo [Brézillon et al., 2008]. Destacan que casi todos los esfuerzos se han centrado en elementos físicos del contexto como la hora, el clima y otro tipo de información que puede ser obtenida a través de sensores y ser usada directamente en aplicaciones. Más aun, indican que generalmente se considera el contexto de un solo usuario, no el contexto de un grupo. Estas observaciones están estrechamente relacionadas con la problemática atacada y con los objetivos planteados en la presente tesis de maestría.

Sin embargo, la definición de contexto de uso propuesta por Brézillon et al. se centra principalmente en el "conocimiento grupal", i.e., en cómo analizar los datos físicos, sociales e históricos que cada usuario aporta y a partir de ellos obtener el estado general de la comunidad.

Síntesis

Se destacan los siguientes puntos del análisis previo sobre el concepto de contexto de uso:

- de acuerdo a la literatura científica consultada, se han hecho muy pocos estudios sobre el contexto de uso en el ámbito de los sistemas colaborativos. Además ninguno de ellos se ha enfocado en el entorno social en donde se lleva a cabo la interacción entre los colaboradores;
- las variables más documentadas son aquellas cuya fuente proviene de sensores físicos, dejando en segundo lugar a las variables lógicas, las cuales son relevantes si se desea describir el contexto en términos de roles sociales, preferencias y habilidades de los colaboradores;
- casi todos los autores coinciden en indicar que el contexto debe ser descrito con respecto a entidades conocidas como el usuario, el sistema, la ubicación o la tarea a realizar. En el caso de los sistemas colaborativos, las entidades serían las mismas, con la única diferencia que se consideraría el contexto de múltiples usuarios, proyectos y ubicaciones.

El presente trabajo de tesis toma como punto de partida la definición de contexto proporcionada por Coutaz y Rey [Coutaz and Rey, 2002], la cual considera al contexto como la composición de multiples situaciones en un cierto periodo de tiempo. En la mayoría de los sistemas mono-usuario, todas las situaciones capaces de ser detectadas afectan a la misma entidad: el usuario. En cambio, en los sistemas colaborativos se debe tomar en cuenta un mayor número de entidades (e.g., colaboradores, proyectos, tareas y ubicaciones), cada una afectada por diferentes situaciones. Por lo tanto, cuando nos referimos al contexto en entornos colaborativos, en realidad hablamos del **contexto de cada entidad presente**.

Una consecuencia de considerar un mayor número de entidades es que el número de variables involucradas aumenta también, dando lugar a situaciones más complejas.

2.2 Sistemas sensibles al contexto de uso

Dado que el objetivo de la presente tesis es proponer una arquitectura contextual que permita detectar y considerar cambios en el contexto de uso en sistemas colaborativos, es necesario conocer algunos de los sistemas y aplicaciones sensibles al contexto representativos.

Los sistemas sensibles al contexto son aquellos capaces de adaptar su funcionamiento al contexto actual sin la intervención explícita del usuario, con el objetivo de incrementar la usabilidad y efectividad del sistema [Baldauf et al., 2007].

A continuación se describe de forma general algunos sistemas adaptativos relevantes. Después de proporcionar una síntesis de cada uno de los sistemas analizados se hace un estudio comparativo.

2.2.1 Call Forwarding

Corresponde a una de las primeras aplicaciones contextuales reportadas en la literatura científica. Particularmente, se trata de la aplicación de demostración para el sistema de ubicación Active Badge [Want et al., 1992]. Esta aplicación proporciona información a un recepcionista sobre la ubicación del personal, facilitando así la tarea de redireccionar las llamadas al lugar más cercano en donde sus destinatarios se encuentren. La información proporcionada por el sistema se muestra en la figura 2.3 y se refiere a los nombres de las personas junto con los campos de la ubicación, la extensión telefónica más cercana y la probabilidad, en forma de porcentaje, de encontrar a una persona en el lugar indicado. Si el porcentaje es menor al 100 % significa que la persona se está moviendo pero si no ha sido vista en más de una semana se despliega la etiqueta 'AWAY'. El recepcionista redirecciona las llamadas telefónicas con base en la información mostrada.

Los autores reportan que la aplicación resultó ser útil tanto para el personal como para el recepcionista, aunque se detectaron algunos inconvenientes debido a que no se consideró el control de situaciones en las que no se deseaba recibir llamadas. Por ejemplo, la mayoría de las personas no desean recibir llamadas cuando están en la oficina de su jefe o en el comedor entre la 1:00 y 2:00 P.M.

2.2.2 Ubidraw

UbiDraw [Vanderdonckt and González-Calleros, 2008] es una aplicación de dibujo vectorial que adapta su interfaz gráfica desplegando, ocultando, redimensionando y reacomodando las barras de herramientas e iconos de acuerdo a la tarea actual que el usuario realiza, la frecuencia de uso de las herramientas, las preferencias detectadas y el espacio disponible. En este sistema se monitorea el estado de todas estas variables

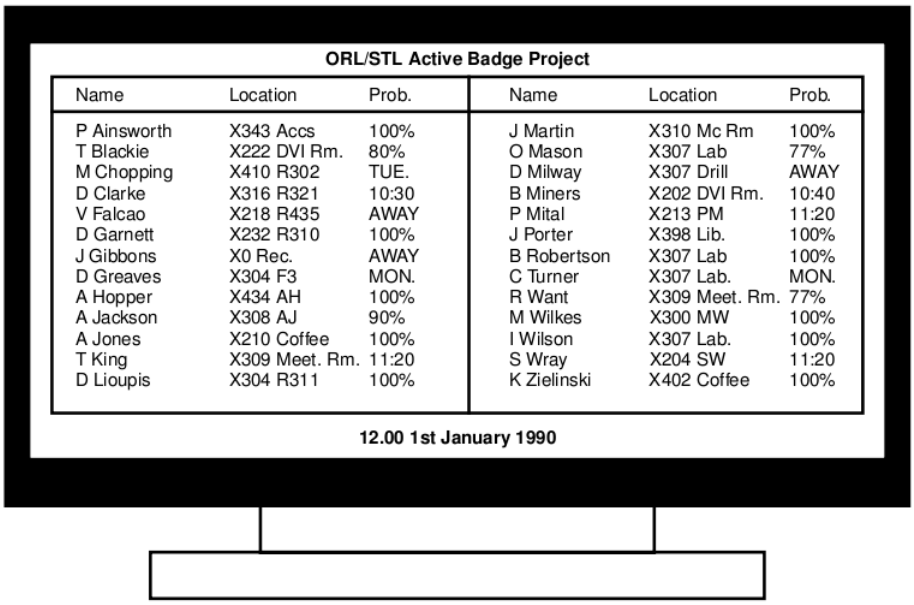


Figura 2.3: Información presentada por el sistema de rastreo de llamadas Call Forwarding

para maximizar el número de herramientas mostradas. La figura 2.4 muestra un ejemplo de la adaptación del sistema ante el redimensionamiento de la ventana.

UbiDraw basa su funcionamiento en un componente denominado observador contextual (*ContextWatcher*) el cual monitorea las tareas realizadas por el usuario y envía esta información a un conjunto de *widgets* ubicuos (*UbiWidgets*) que son componentes plásticos de la interfaz del usuario. La adaptación en UbiDraw siempre es resultado de la iniciativa del usuario.

2.2.3 UbiCicero

UbiCicero [Ghiani et al., 2009] es una guía de museo sensible a la ubicación. El sistema funciona sobre dispositivos móviles equipados con un lector RFID que detecta las obras de arte previamente etiquetadas (cf. figura 2.5). El objetivo principal de este sistema es proveer a los usuarios información y servicios en función de su ubicación.

Las etiquetas colocadas en las obras de arte contienen un identificador único. Una sola etiqueta puede estar asociada a varias obras de arte cercanas. Cuando los usuarios entran a un nuevo cuarto se les muestra primeramente un breve resumen de la sección previamente visitada y posteriormente el mapa del nuevo cuarto. Cuando el usuario se acerca a una obra de arte se le pregunta si desea mayor información, la cual es proporcionada en caso de obtener una respuesta afirmativa.

El sistema también soporta la interacción entre varios usuarios mediante juegos cooperativos (en los cuales los usuarios se dividen en equipos). En el mapa de los

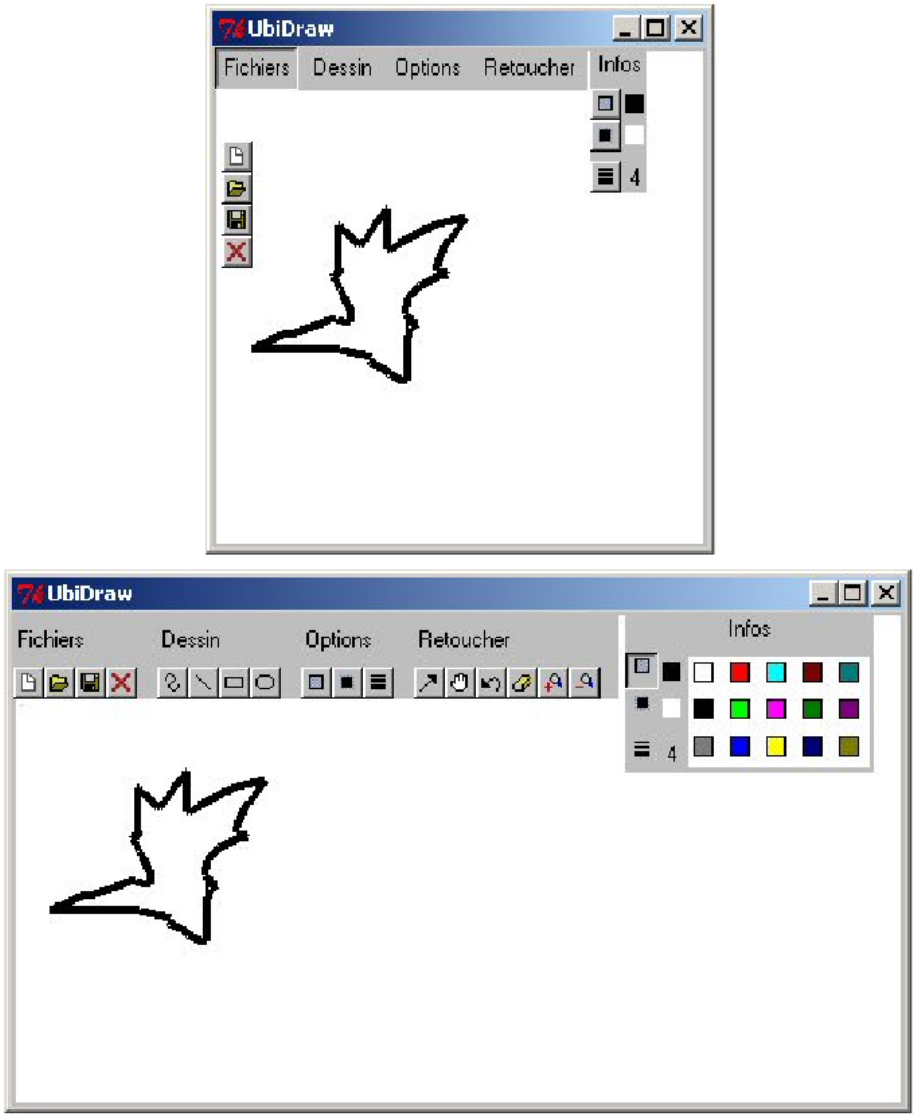


Figura 2.4: Aplicación UbiDraw: los íconos mostrados y su ubicación dependen de la tarea actual del usuario, de la frecuencia de uso y del espacio disponible en la ventana

Figura 2.5: UbiCicero, una guía de museo sensible a la ubicación del usuario: el sistema se ha montado en una PDA equipada con un lector RFID

cuartos del museo, se identifican con colores distintivos las obras de arte que han sido visitadas por otros jugadores del mismo equipo. Aparte de los dispositivos móviles, el sistema considera la participación de despliegues públicos, en los cuales se visualiza el mapa completo del museo con la ubicación de cada usuario y de las obras de arte que se han visitado.

A continuación se describe brevemente los módulos de la arquitectura de UbiCicero:

- **Definición del museo:** provee las especificaciones del museo, tales como dimensiones de los cuartos, obras de arte, recursos e información multimedia.
- **Ubicación:** detecta la ubicación de los usuarios mediante tecnología RFID.
- **Núcleo:** proporciona las rutinas de comunicación con los dispositivos móviles y despliegues públicos.
- **Usuario:** controla las preferencias individuales de cada usuario.
- **Visita:** soporta la presentación de los mapas y elementos interactivos.
- **Juegos:** describe los juegos disponibles mediante representaciones XML.

2.2.4 Sistema de control de calefacción para casas

El sistema de control de calefacción para casas [Calvary et al., 2001] es un sistema propuesto por EDF (*French Electricity Company*) que permite controlar la calefacción de distintas habitaciones a través de diferentes contextos de uso, tales como:

- En casa, por medio de un dispositivo específico montado en la pared o a través de una PDA conectada a la red inalámbrica de la casa.
- En la oficina, mediante la Web, usando una PC estándar.
- Desde cualquier lugar, usando un teléfono celular equipado con WAP (*Wireless Application Protocol*).

La interfaz gráfica de este sistema se adapta al dispositivo de despliegue. En una PC se muestra al mismo tiempo la temperatura de todos los cuartos (cf. figura 2.6.a), mientras que en una PDA la temperatura de cada cuarto se muestra en pestañas diferentes (cf. figura 2.6.b) y en un celular se hace una remodelación completa de la interfaz para poder adaptarse al reducido espacio y al tipo de modalidad disponible: textual en lugar de gráfica (cf. figura 2.6.c).

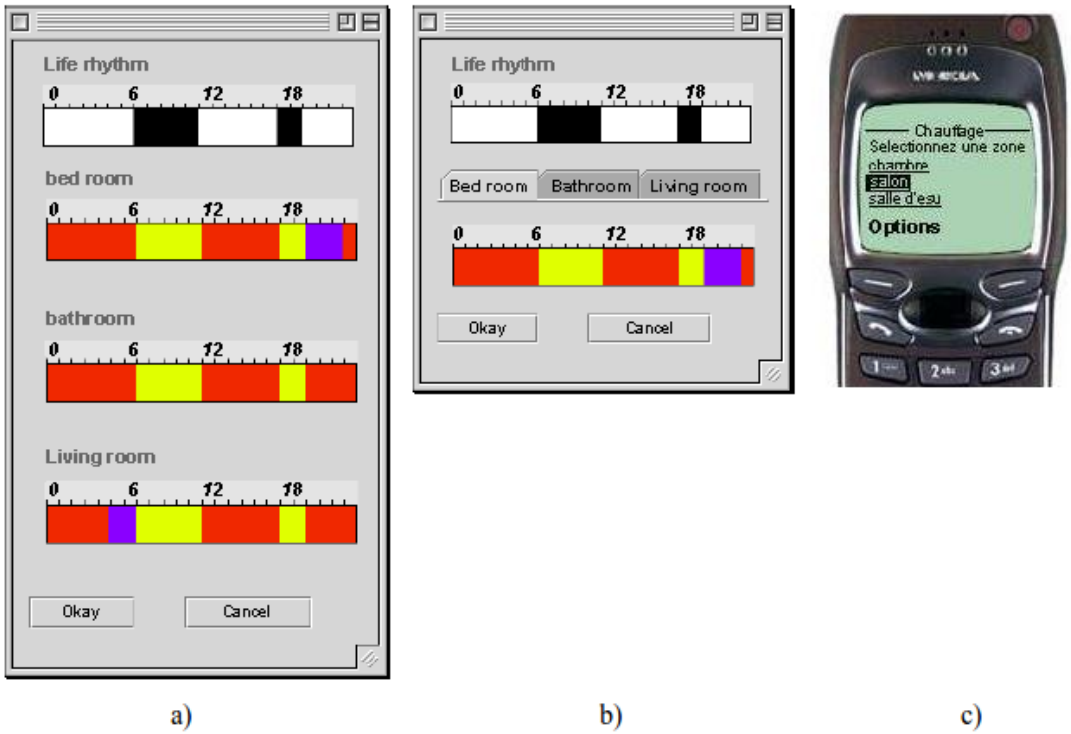


Figura 2.6: El sistema de control de calefacción para casas cuya interfaz gráfica se adapta al dispositivo de despliegue: a) en una PC se muestran todos los cuartos al mismo tiempo; b) en una PDA la temperatura de cada cuarto se muestra en pestañas diferentes; c) la interfaz gráfica se modifica completamente para adaptarse al reducido tamaño de un celular y al tipo de modalidad disponible: textual en lugar de gráfica

Este sistema fue creado a partir de la herramienta ARTStudio[1], diseñada por Calvary et al. con el propósito de soportar varios de los conceptos que proponen en su *framework* de unificación para el desarrollo de interfaces de usuario plásticas [Calvary et al., 2001].

2.2.5 Conference Assistant

Conference Assistant [Dey et al., 1999] es una aplicación diseñada para proporcionar información útil a las personas que asisten a una conferencia en la que se planea diferentes actividades como presentaciones de artículos, demostraciones, reuniones, etc.

Una descripción típica del escenario de uso de esta aplicación es la siguiente: los usuarios que asisten a una conferencia se registran proporcionando sus datos, información de contacto y una lista de intereses. De la misma manera, ellos reciben la

[1] Adaptation by Reification and Translation

aplicación Conference Assistant, la cual puede ser ejecutada en su computadora personal. La aplicación despliega los horarios y actividades de la conferencia, destacando aquellas actividades que pueden ser de mayor interés para cada usuario con base en la lista de intereses proporcionada.

Cuando un usuario entra a un cuarto, la aplicación despliega automáticamente el nombre del presentador, el título de la presentación y otra información relevante. Por otra parte, el usuario es notificado de las actividades que sus colegas cercanos están atendiendo. Conference Assistant hace uso de múltiples variables contextuales: los intereses personales de cada usuario, la ubicación de los asistentes y los detalles de las presentaciones.

2.2.6 Nokia Situations

Nokia Situations [NokiaBetaLabs, 2010] es una aplicación móvil desarrollada por Nokia Beta Labs que se encontraba en fase de pruebas al momento de escribir este documento. Esta aplicación tiene el propósito de adaptar el comportamiento del teléfono celular de acuerdo al contexto, el cual es detectado tomando en consideración las situaciones definidas por el usuario.

Estas situaciones particulares se definen a partir de un conjunto de reglas introducidas manualmente, basadas en los sensores internos del teléfono. Una situación puede ser definida de acuerdo a la hora, al día, a la ubicación GPS y a la disponibilidad de redes WiFi o de dispositivos Bluetooth (cf. figura 2.7).

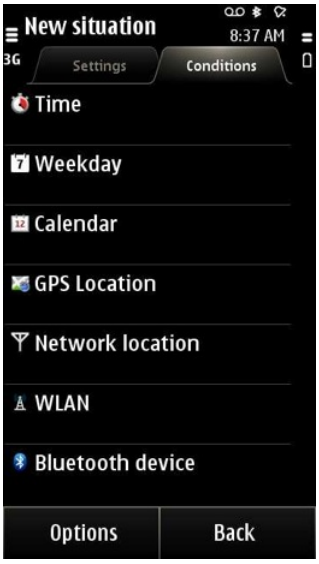


Figura 2.7: Nokia Situations: el usuario puede definir nuevas situaciones en función de varios parámetros disponibles, como la ubicación GPS o la hora

Una vez definidas las situaciones, el usuario configura el comportamiento deseado en el teléfono móvil. El usuario puede definir un tono y volumen para cada situación, ajustar el estado de la alerta vibrante, establecer una respuesta automática a mensajes SMS, cambiar al modo de ahorro de energía (desactivando funciones como Bluetooth) e incluso iniciar automáticamente una aplicación en respuesta al contexto detectado.

2.2.7 Análisis comparativo

Después de haber descrito algunos de los sistemas sensibles al contexto más representativos, se presenta un análisis comparativo de acuerdo a los siguientes parámetros:

1. Sistemas mono-usuario vs. multi-usuario
2. Sistemas enfocados al contexto mono-entidad o multi-entidad
3. Variables contextuales consideradas por los sistemas adaptativos
4. *Framework* de desarrollo o arquitectura específica
5. Tipo de adaptación

Sistemas mono-usuario vs. multi-usuario

La interacción soportada por los sistemas mono-usuario es de tipo usuario-sistema, por lo que el único tipo de retroalimentación proviene de las acciones propias. Por el contrario, los sistemas multi-usuario soportan la interacción usuario-sistema-usuario, lo que implica que el usuario no solo está consciente de sus propias acciones, sino también de las acciones de los demás. La relación entre los usuarios generalmente es de colaboración, sin embargo también puede ser de competencia, e.g., algunos videojuegos.

En la tabla 2.1 se examina si la naturaleza de los sistemas expuestos es mono-usuario o multi-usuario. Call Forwarding es un sistema mono-usuario porque la ubicación de las personas solo es mostrada a un recepcionista. Al igual que la mayoría de las herramientas de dibujo, UbiDraw también es mono-usuario. El sistema de calefacción igualmente es mono-usuario porque no considera el caso en el que al menos dos usuarios participen en la modificación de la temperatura de alguna habitación. Así mismo, la aplicación de Nokia Situations es mono-usuario porque únicamente considera la participación de un solo usuario en la definición de situaciones. En el caso de UbiCicero, la interacción se realiza a partir de juegos y mostrando las obras de arte visitadas por los integrantes del equipo. En la aplicación de Conference Assistant, los usuarios tienen conciencia de la ubicación de sus colegas y de las actividades que están atendiendo. Por lo tanto, UbiCicero y Conference Assistant son sistemas multi-usuario.

Sistema	Mono-usuario	Multi-usuario
Call Forwarding	✓	
UbiDraw	✓	
UbiCicero		✓
Sistema de calefacción	✓	
Conference Assistant		✓
Nokia Situations	✓	

Tabla 2.1: Sistemas mono-usuario vs multi-usuario

Sistemas enfocados al contexto mono-entidad o multi-entidad

Nos referimos a sistemas contextuales mono-entidad cuando solo se considera la situación de una sola entidad, e.g., un usuario en particular o un dispositivo. Mientras que en los sistemas contextuales multi-entidad se considera la situación de varias entidades de forma simultánea, e.g., los miembros de una organización. La tabla 2.2 muestra los sistemas presentados de acuerdo a esta clasificación.

Las aplicaciones Call Forwarding, UbiCicero y Conference Assistant pertenecen a la categoría de sistemas contextuales multi-entidad, debido a que consideran la ubicación de múltiples usuarios y el estado de otras entidades como las obras de arte visitadas en un museo o las actividades disponibles en una conferencia. UbiDraw, por el contrario, es un sistema contextual mono-entidad ya que solo considera el estado del usuario (manifestado a través de su tarea actual y la frecuencia con que usa las herramientas). El sistema de calefacción también es mono-entidad porque solo toma en cuenta el estado de la plataforma, en específico la pantalla de despliegue. Nokia Situations considera múltiples variables contextuales, pero todas con respecto a un solo teléfono celular, por lo tanto también se trata de un sistema contextual mono-entidad.

Sistema	Mono-entidad	Multi-entidad
Call Forwarding		✓
UbiDraw	✓	
UbiCicero		✓
Sistema de calefacción	✓	
Conference Assistant		✓
Nokia Situations	✓	

Tabla 2.2: Sistemas contextuales mono-entidad vs multi-entidad

Variables contextuales consideradas por los sistemas analizados

La tabla 2.3 muestra las variables contextuales con base en las cuales los sistemas se adaptan. La mayoría de los sistemas se adaptan a variables físicas como la ubicación y la hora; solo algunos como Conference Assistant y UbiDraw toman en cuenta variables lógicas como los intereses personales de los usuarios o la frecuencia de uso de las herramientas, respectivamente.

Sistema	Variables contextuales
Call Forwarding	Ubicación de las personas
UbiDraw	Tarea actual del usuario, frecuencia de uso de las herramientas y tamaño de la ventana
UbiCicero	Ubicación de los usuarios y obras de arte detectadas
Sistema de calefacción	Plataforma
Conference Assistant	Intereses personales de los usuarios, ubicación de los asistentes y detalles de las presentaciones
Nokia Situations	Hora, día, ubicación GPS, disponibilidad de redes Wi-Fi y de dispositivos Bluetooth

Tabla 2.3: Variables contextuales consideradas por los sistemas analizados

Framework de desarrollo o arquitectura específica

En esta clasificación se indica si el sistema fue creado a partir de un *framework* de desarrollo o si se deriva de una arquitectura creada específicamente para el sistema en cuestión.

En la tabla 2.4 se puede visualizar que el desarrollo de la mayoría de los sistemas parte desde cero, ya que se basan en el diseño de arquitecturas específicas. Dentro de los sistemas estudiados, solo el sistema de calefacción fue construido a partir del *framework* ARTStudio [Calvary et al., 2001]. Por otra parte, los creadores de Nokia Situations no especifican la arquitectura usada.

Tipo de adaptación

Según Abowd et al., la adaptación de los sistemas sensibles al contexto se puede manifestar de las siguientes maneras [Abowd et al., 1999]:

- **Presentación de información y servicios al usuario:** se refiere a una técnica de interacción en donde se presenta una lista de objetos (e.g., impresoras) o

Sistema	Arquitectura dedicada	Framework	No específica
Call Forwarding	✓		
UbiDraw	✓		
UbiCicero	✓		
Sistema de calefacción		✓	
Conference Assistant	✓		
Nokia Situations			✓

Tabla 2.4: Sistemas clasificados de acuerdo a si fueron creados con un *framework* o bajo una arquitectura específica

lugares (e.g., oficinas) cuyos elementos más relevantes, de acuerdo al contexto de uso, son enfátizados o más fáciles de escoger.

- **Ejecución automática de un servicio:** en este caso no solo se presenta información relevante sino que, dada la correcta combinación de condiciones contextuales, se ejecuta automáticamente una acción.

- **Información aumentada:** algunos datos contextuales pueden ayudar a comprender mejor el entorno colaborativo en el que se trabaja, e.g., se puede identificar bajo qué situaciones las personas son más activas.

La tabla 2.5 muestra la clasificación de los sistemas presentados de acuerdo a su tipo de adaptación.

Sistema	Presentación	Ejecución	Aumentación
Call Forwarding			✓
UbiDraw	✓		
UbiCicero		✓	✓
Sistema de calefacción	✓		
Conference Assistant	✓		✓
Nokia Situations		✓	

Tabla 2.5: Clasificación de los sistemas estudiados con base en su tipo de adaptación

La aplicación Call Forwarding solo aumenta información sobre la ubicación de las personas y la extensión telefónica más cercana a ellas, sin embargo no enfatiza nada en específico ni ejecuta automáticamente algún servicio, por esta razón su adaptación solo entra en la categoría de información aumentada.

UbiDraw reacomoda las herramientas enfatizando aquellas que son usadas con más frecuencia, por lo tanto su adaptación se enfoca a la presentación de información y servicios al usuario.

UbiCicero aumenta la información disponible al dar a conocer la ubicación de otros usuarios y las obras de arte que han visitado. Así mismo, al detectar que un usuario ha entrado a un nuevo cuarto o que está cerca de cierta obra de arte, este sistema ejecuta automáticamente el servicio de proporcionar mapas e información relevante. En consecuencia, UbiCicero muestra dos tipos de adaptación: ejecución automática de servicios y aumentación de información.

En el caso de Conference Assistant también da a conocer la ubicación de otros colegas y las actividades que están atendiendo, pero enfatiza las actividades que pueden ser de mayor interés para el usuario. Por lo tanto, la adaptación en Conference Assistant entra en las categorías de información aumentada y presentación de información y servicios al usuario.

La adaptación del sistema de calefacción cae en la categoría de presentación de información y servicios al usuario debido a que, dependiendo del tamaño disponible en pantalla, puede mostrar todos los cuartos simultáneamente o enfatizar aquel por el que el usuario tiene preferencia.

Por último, la adaptación en Nokia Situations es del tipo ejecución automática de servicios, pues una vez detectada una situación se ejecuta automáticamente el cambio correspondiente en la configuración del teléfono. No se enfatizan elementos, ni se busca aumentar la información.

Síntesis del análisis comparativo

Se destacan los siguientes puntos:

- La mayoría de los sistemas adaptativos son sistemas mono-usuario.
- La ubicación es la variable contextual que más se considera en estos sistemas.
- El desarrollo de la mayoría de los sistemas que se adaptan al contexto se realiza desde cero con una arquitectura específica al problema a resolver.
- No se favorece algún tipo de adaptación en específico; generalmente la adaptación es una mezcla de varias acciones que incluyen presentación, ejecución automática y aumentación de la información.

2.3 *Frameworks* para el desarrollo de sistemas contextuales

Mucho trabajo de investigación sobre arquitecturas y *frameworks* de desarrollo se enfoca en el diseño de sistemas contextuales. Algunos de ellos se centran en aspectos particulares, como el desarrollo de interfaces de usuario capaces de adaptarse al contexto de uso actual.

Por ejemplo, Calvary et al., proponen un *framework* conceptual que intenta unificar el proceso de desarrollo de interfaces de usuario plásticas, capaces de adaptarse en función de las variaciones del contexto de uso, preservando la usabilidad del sistema [Calvary et al., 2001]. En tal *framework* se proporciona métricas, costos y procesos de adaptación que pueden ser tomados como guía en el desarrollo de interfaces de usuario plásticas.

El contexto de uso juega un papel importante en la creación y organización de ambientes y espacios inteligentes. La arquitectura CoBrA (Context Broker Architecture) propuesta por Chen se enfoca en Cómputo Pervasivo [Chen, 2004]. Las mayores contribuciones de su trabajo incluyen la definición de una arquitectura basada en un agente central para el soporte de sistemas contextuales y una ontología para Cómputo Pervasivo en ambientes generalizados.

La mayoría de estos *frameworks* considera el contexto de un único usuario; mientras que el contexto de un grupo colaborativo solo recientemente se ha tomado en cuenta [Brézillon et al., 2008], por lo que permanecen abiertas algunas cuestiones importantes, como la forma de combinar los estados lógicos y físicos de los miembros del grupo y el determinar qué tipo de situaciones pueden ser explotadas.

Sendín et al. describen un *framework* conceptual cuyo objetivo es servir como referencia para la generación de interfaces de usuario plásticas para ambientes colaborativos, tanto en el proceso de plasticidad explícita[2] [Sendín et al., 2008] como en el de plasticidad implícita[3] [Sendín and Collazos, 2006]. Particularmente, se toma como variables contextuales a la conciencia de grupo y al conocimiento compartido. A pesar de que su trabajo considera el contexto de un grupo colaborativo, su principal limitación es que permanece como una propuesta teórica que no se evalúa frente a escenarios de la vida real.

Cardoso y José proponen un *framework* para la adaptación del contenido mostrado en un despliegue público con base en el contexto presente [Cardoso and José, 2009]. Las variables contextuales que se consideran son la detección de individuos cercanos, la caracterización del público con base en su edad o género, la identificación de las personas, las sugerencias hechas por los usuarios, la detección de la reacción del público al contenido desplegado, etc. La principal contribución de su investigación radica en el análisis de las variables que se consideran y en las propuestas de cómo explotarlas. A pesar de que los autores consideran un gran número de personas, los grupos colaborativos están fuera del alcance de su trabajo.

Hussein et al. describen una arquitectura dirigida a sistemas contextuales de recomendación grupal [Hussein et al., 2010]. En su trabajo, las recomendaciones se forman tomando en cuenta variables físicas como la ubicación y el clima, así como variables lógicas como el historial de navegación y el contenido descargado. Sin embargo, este *framework* se enfoca en sistemas de recomendación, por lo tanto no es genérico.

[2]El cliente explícitamente solicita al servidor los cambios que debe hacer para adaptarse al contexto de uso.

[3]El cliente reacciona ante los cambios del contexto de uso sin la intervención del servidor.

Finalmente, Haake et al. [Haake et al., 2010] proponen un *framework* genérico de cuatro capas para modelar y explotar el contexto en sistemas adaptativos colaborativos. Los autores abordan el problema utilizando un enfoque ontológico. Resultados importantes se reportan frente a cuatro problemas típicos en ambientes colaborativos: *co-ubicación* (denota la situación en la que varios usuarios se encuentran en el mismo lugar físico o virtual), *co-acceso* (cuando multiples personas acceden al mismo artefacto), *co-recomendación* (se refiere a la situación en la que acciones implícitas o explícitas de los colaboradores se usan para recomendar información o recursos a otros colaboradores) y *co-dependencia* (situación en la que existe dependencia entre tareas, objetos y usuarios). Sin embargo, su trabajo principalmente se enfoca en la descripción de la ontología (restringida a un modelo específico) y no provee detalles sobre su integración con un sistema colaborativo real.

Síntesis

Se destacan los siguientes puntos en los *frameworks* de desarrollo analizados:

- La mayoría de las arquitecturas y *frameworks* de desarrollo para sistemas que se adaptan al contexto son propuestas meramente conceptuales.
- Los *frameworks* generalmente se enfocan en sistemas mono-usuario. Solo recientemente se ha propuesto *frameworks* que contemplan a sistemas multi-usuario y colaborativos.
- Principalmente se consideran variables físicas como la ubicación y la hora, solo unos cuantos trabajos toman en cuenta otras variables que son propias de los grupos de trabajo colaborativos.
- Pocos trabajos aportan detalles sobre implementación, mientras que otros no consideran casos de uso ni análisis de escenarios reales, i.e., no existe un mecanismo de validación de las ideas propuestas.

Capítulo 3

Análisis y diseño

En el presente capítulo se analizan algunas situaciones comunes en grupos de trabajo y se dan algunas sugerencias de como aprovecharlas para mejorar la usabilidad y eficiencia de los sistemas colaborativos (cf. sección 3.1). Algunos de estos escenarios se retoman de trabajos relacionados previos, pero también se incluyen propuestas que no han sido estudiadas con anterioridad.

Las situaciones analizadas conforman la base del diseño de la arquitectura contextual propuesta, la cual consiste de cuatro fases: 1) percepción de eventos contextuales, 2) detección de situaciones, 3) comunicación de situaciones y 4) adaptación del sistema colaborativo. Detalles y consideraciones de cada fase se proporcionan también en este capítulo (cf. sección 3.2).

3.1 Análisis de casos de uso

En esta sección se describen algunas situaciones contextuales que pueden ser explotadas en sistemas colaborativos. Mientras que algunas de estas situaciones se retoman de la literatura científica existente (cf. subsección 3.1.2), otras son analizadas por primera vez (cf. subsección 3.1.1).

3.1.1 Situaciones propuestas

Las siguientes situaciones contextuales no han sido analizadas previamente en la literatura científica, sin embargo son comunes en entornos colaborativos. Se proponen ideas de cómo aprovechar estas situaciones al ser detectadas en sistemas colaborativos.

Periodo de introducción

Cuando un nuevo colaborador se integra a un equipo de trabajo, comúnmente invierte tiempo en adquirir la información necesaria para adaptarse al grupo. Por ejemplo, debe conocer las políticas internas de la organización, los procedimientos, los reglamentos e información de los colaboradores con los que tendrá contacto.

El sistema colaborativo puede reducir el tiempo requerido para que el nuevo colaborador adquiera los conocimientos necesarios. Al detectar que el usuario recientemente se incorporó al equipo, puede mostrar periódicamente la información relevante que necesita (cf. figura 3.1).

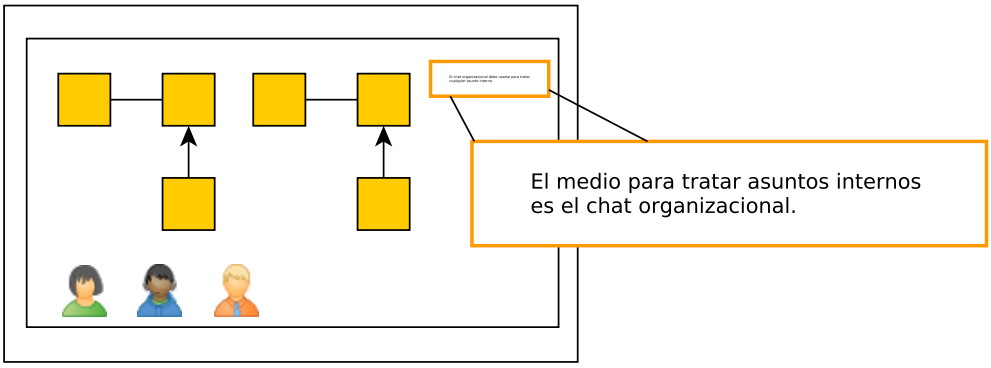


Figura 3.1: El sistema colaborativo muestra periódicamente información relevante a los usuarios en periodo de introducción

Hora de salida

La elaboración de reportes para comunicar las actividades y horas de trabajo es una práctica común y muchas veces obligatoria en las organizaciones. Es una actividad importante porque permite comparar los tiempos planeados contra los tiempos reales en la realización de un proyecto. Generalmente, esta práctica se realiza al terminar las actividades del día correspondiente.

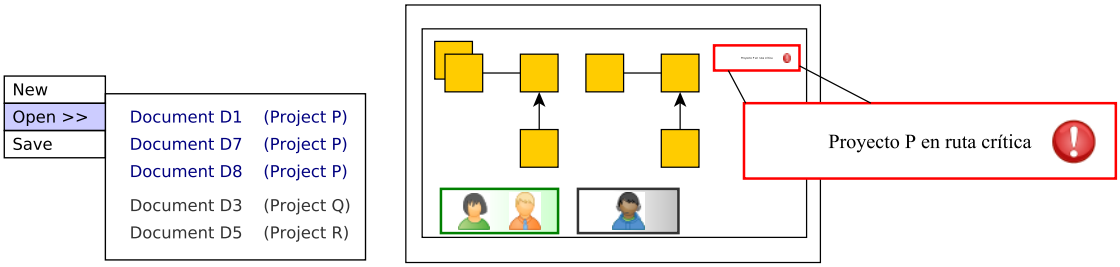
Un sistema contextual podría detectar que la hora de salida está próxima (o que ya se ha trabajado suficiente tiempo) y podría ejecutar automáticamente las aplicaciones y herramientas necesarias para que los usuarios reporten sus actividades realizadas. El usuario ya no sería responsable de abrir la aplicación por sí mismo, por lo que se reduce el riesgo de olvidar reportar sus actividades.

Algunas otras tareas que se pueden hacer al final del día son el respaldo y la integración de información. Estas tareas se pueden realizar automáticamente con un sistema contextual que detecte el evento correspondiente.

Proyecto en ruta crítica

En administración de proyectos, la ruta crítica es la secuencia de los elementos terminales de la red de proyectos con la mayor duración entre ellos, determinando el tiempo más corto en el que es posible completar el proyecto. La duración de la ruta crítica determina la duración del proyecto entero. Cualquier retraso en un elemento de la ruta crítica afecta a la fecha de término planeada del proyecto.

Un sistema colaborativo que detecte que el proyecto sobre el que se trabaja está en ruta crítica puede adaptarse para no desplegar distractores, facilitar el acceso a los documentos relacionados, bloquear o deshabilitar documentos no relacionados (cf. figura 3.2a). Incluso se puede realizar un reordenamiento en los elementos de conciencia de grupo, e.g., resaltar las fotos de las personas relacionadas con el proyecto, con el fin de agilizar la comunicación entre los miembros del equipo involucrados en el mismo proyecto (cf. figura 3.2b).



(a) Reordenamiento de los documentos, facilitando el acceso a los documentos relacionados y/o deshabilitando los no relacionados

(b) Reordenamiento de los elementos de la barra de colaboradores, resaltando a las personas involucradas en el proyecto

Figura 3.2: Propuestas de adaptación de un sistema colaborativo al detectar que el proyecto P está en ruta crítica

Colaborador ausente

Ante la situación de que uno de los colaboradores se detecte como ausente y se determine que su presencia es necesaria para la realización de alguna actividad, el sistema colaborativo puede adaptarse para: 1) desplegar alternativas de comunicación con el objetivo de contactar al colaborador ausente lo más pronto posible o 2) mostrar una lista de los colaboradores con habilidades similares para que cubran las tareas del colaborador ausente (cf. figura 3.3).

La notificación de esta situación solo es de interés para las personas con los roles adecuados para tomar decisiones sobre lo que debe hacerse para cubrir la ausencia del

colaborador. Por lo tanto, la adaptación del sistema colaborativo no se realiza para todos los colaboradores, sino solo para aquellos con el rol adecuado.

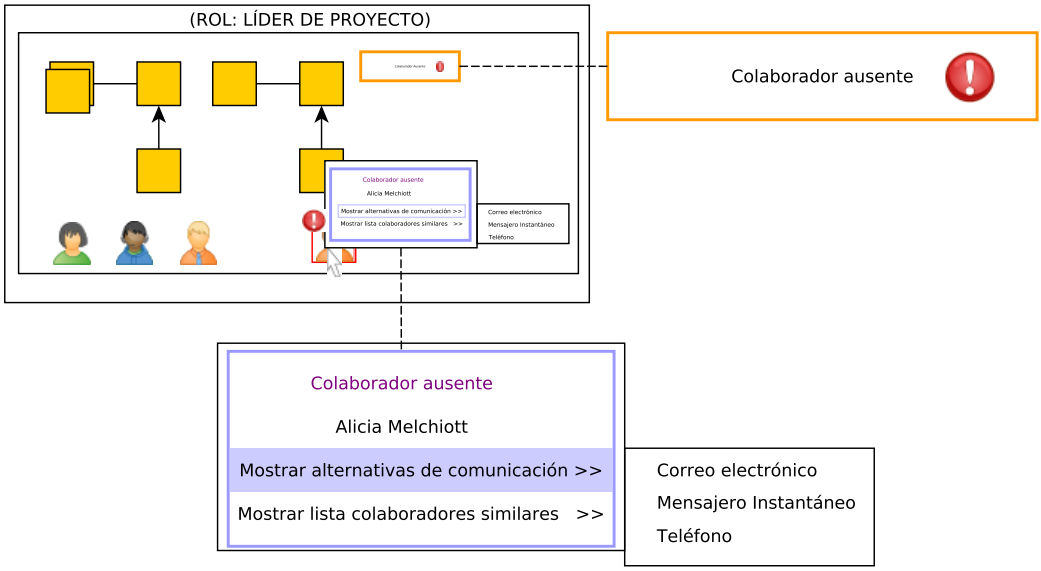


Figura 3.3: El sistema colaborativo muestra diferentes opciones ante la ausencia de un colaborador

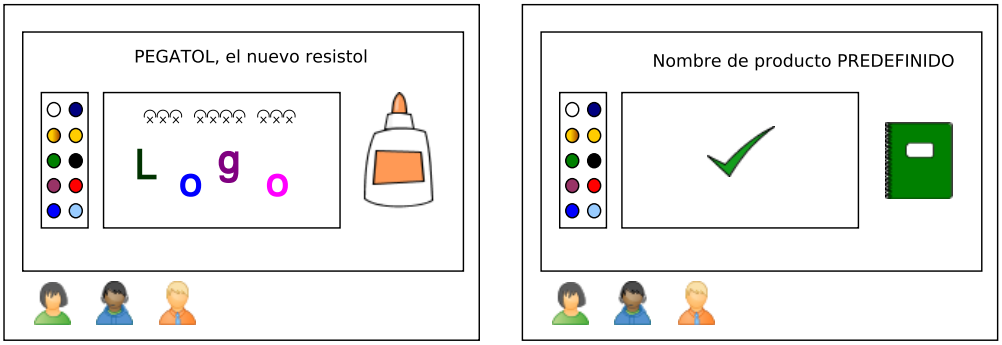
Persona externa detectada

La información con la que se trabaja en algunos proyectos puede ser confidencial, por lo que solo debe ser conocida por los colaboradores que trabajan en el proyecto. La detección de una persona ajena al equipo de colaboración puede traducirse en la adaptación del sistema colaborativo para ocultar información sensible. Observe el siguiente escenario:

"Suponga que un grupo de colaboradores trabaja en el diseño de un logotipo para un nuevo producto que un cliente está a punto de lanzar al mercado. Todos se encuentran reunidos en una sala de juntas. En un momento dado, una persona entra a la oficina. El sistema colaborativo que están usando para alcanzar su objetivo detecta que la persona que acaba de entrar es externa al equipo de colaboración, por lo que se encarga de ocultar (o alterar) la información sensible, e.g., nombre del producto, imagen y logotipo. Cuando la persona externa sale de la sala de juntas, la información se despliega de nuevo (cf. figura 3.4)".

Reunión

Si un sistema colaborativo detecta esta situación puede adaptarse para facilitar el acceso a documentos relacionados con el proyecto sobre el que se trabaja (cf. figura



(a) El nombre del producto, la imágen y logotipo se muestran normalmente

(b) Al detectarse una persona externa se oculta información sensible, reemplazándose por información predefinida

Figura 3.4: Adaptación de un sistema colaborativo ante la detección de una persona externa

3.5). Así mismo puede bloquear los mensajes no relevantes hasta que la reunión termine.

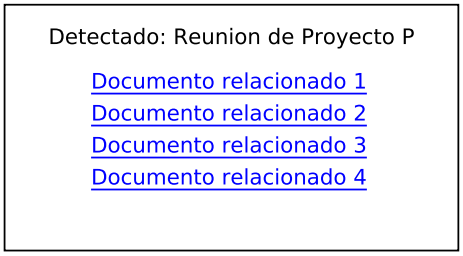


Figura 3.5: El sistema colaborativo se adapta mostrando los documentos relevantes a la reunión detectada.

3.1.2 Situaciones citadas en la literatura científica

Algunas situaciones de la literatura científica fueron retomadas por este trabajo con la finalidad de diseñar una arquitectura general e independiente de variables contextuales y situaciones específicas.

Tazas de café

Coutaz y Rey presentan un escenario basado en una experiencia de la vida real: “En nuestro laboratorio, las tazas de café son un recurso compartido. Por lo tanto deberían mantenerse en la cafetería todo el tiempo. Sin embargo frecuentemente tomamos el

café en la oficina y olvidamos regresar las tazas a la ubicación apropiada. Como resultado, la primera persona sin taza de café disponible, envía mensajes solicitando que devuelvan las tazas. Para evitar esta situación, se implementó un sistema destinado a enviar automáticamente un mensaje de alerta por correo electrónico cuando el número de tazas en la cafetería está por debajo de un umbral [Coutaz and Rey, 2002]".

En el escenario descrito, la variable contextual involucrada es la cantidad de tazas disponibles (cf. figura 3.6). El sistema mencionado por Coutaz y Rey no es necesariamente un sistema colaborativo, sin embargo se analiza con el fin de cubrir el requerimiento de generalidad, ya que la arquitectura contextual que se propone en este documento de tesis no se restringe estrictamente a sistemas colaborativos.

Figura 3.6: Sistema contextual que envía correos electrónicos con base en el número de tazas disponibles

Ambiente seguro

Sumayah Al-Rwais y Jalal Al-Muhtadi describen una situación en la que es importante verificar que se cumplan ciertas condiciones al manejar información sensible, e.g., documentos secretos, los cuales antes de ser desplegados en un proyector durante una reunión es necesario tomar en cuenta el contexto ambiental. Por ejemplo esos documentos pueden ser abiertos (i.e., el acceso es acceso otorgado) solo si todas o algunas de las siguientes condiciones se cumplen [Al-Rwais and Al-Muhtadi, 2009]:

a) todo el personal presente está autorizado

b) una reunión está programada

c) la petición de acceso se realiza durante horas de oficina

d) se encuentra presente una persona con autoridad necesaria

e) la puerta de la habitación está cerrada

La situación descrita se relaciona con Cómputo Pervasivo, área de los sistemas computacionales en donde el contexto juega un rol importante. La arquitectura contextual que se propone en el presente documento también cubre este tipo de situaciones.

3.2 Diseño de la arquitectura contextual

En esta sección se describe el diseño de la arquitectura contextual para soportar la detección de cambios en el contexto de uso, con el objetivo de que los sistemas computacionales se adapten a ellos. Dicha arquitectura está dirigida particularmente a sistemas colaborativos (*groupware*), i.e., sistemas que soportan la colaboración de múltiples usuarios. Por lo tanto, se toman en cuenta variables contextuales como el estado de los proyectos, las políticas organizacionales, la ubicación física de los colaboradores, los recursos disponibles y otras variables típicas en los grupos de trabajo. Primeramente se realiza una descripción condensada de la arquitectura (cf. subsección 3.2.1), posteriormente se explica detalladamente cada una de las fases que integran la arquitectura:

1. Percepción de eventos contextuales (cf. subsección 3.2.2)
2. Detección de situaciones (cf. subsección 3.2.3)
3. Comunicación de situaciones (cf. subsección 3.2.4)
4. Adaptación del sistema colaborativo (cf. subsección 3.2.5)

Finalmente se describe como integrar la arquitectura contextual propuesta en un sistema colaborativo típico (cf. subsección 3.2.6).

3.2.1 Descripción general de la arquitectura

La arquitectura contextual consta de cuatro fases: 1) percepción de eventos contextuales, 2) detección de situaciones, 3) comunicación de situaciones y 4) adaptación del sistema (cf. figura 3.7).

De forma breve, cada fase consiste en lo siguiente:

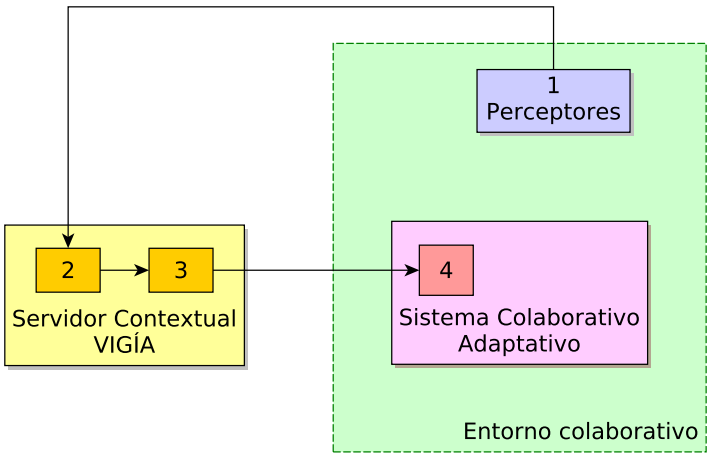


Figura 3.7: Fases de la arquitectura propuesta

1. **Percepción de eventos contextuales:** los entornos colaborativos están sujetos a diferentes variables, las cuales pueden ser físicas (e.g., temperatura, proximidad, detección de personas y dispositivos, niveles de ruido) o lógicas (e.g., estado de los proyectos, prioridad de las actividades, políticas de la organización, roles de usuario). La fase de percepción de eventos contextuales se refiere a la detección de cambios significativos en estas variables, los cuales pueden provocar cambios en el contexto.

2. **Detección de situaciones:** los cambios significativos de las variables contextuales son comunicados a un servidor encargado de determinar si se ha activado una nueva situación (e.g., una reunión en la sala de juntas) o bien, se han invalidado situaciones previas.

3. **Comunicación de situaciones:** mediante el esquema *Publish/Subscribe*, los cambios en las situaciones son notificados a los nodos del sistema colaborativo, interesados en recibir dichas actualizaciones.

4. **Adaptación del sistema:** esta fase se refiere a la adaptación del sistema colaborativo de acuerdo a las situaciones presentes.

3.2.2 Percepción de eventos contextuales

Los entornos colaborativos están sujetos a diversos eventos. Definimos como **eventos contextuales** a aquellos que pueden conducir a la activación o invalidación de situaciones. Algunos ejemplos de eventos contextuales son los siguientes:

- Marco entra a la sala de juntas.
- La temperatura en la oficina de José es de 20 ^{o}C.

- Un proyector ha sido encendido
- Una prórroga a la fecha de entrega del proyecto P ha sido otorgada.
- Carlos se ha incorporado al equipo de trabajo.
- La reunión de mañana ha sido cancelada.

Dichos eventos pueden ser clasificados como eventos contextuales físicos o lógicos.

Eventos contextuales físicos

Reflejan cambios en variables físicas tales como temperatura, proximidad, detección y ubicación de personas y dispositivos, niveles de ruido, etc. En la actualidad existen sensores de hardware capaces de capturar casi cualquier tipo de dato físico [Baldauf et al., 2007]. En la tabla 3.1 se muestran algunos ejemplos de variables físicas con sus respectivos sensores asociados.

Variable física	Sensores disponibles
Luz	Fotodiodos, sensores de color, sensores IR y UV
Audio	Micrófonos
Temperatura	Termómetros, sensores IR
Tacto	Sensores de contacto, interruptores
Distancia	Sensores ultrasónicos, sensores IR
Olor	Sensores electroquímicos

Tabla 3.1: Sensores de hardware comúnmente usados

Sin embargo, para monitorear algunos datos físicos no es suficiente el uso de un solo sensor, sino que es necesario emplear sistemas más complejos. Por ejemplo, para conocer la ubicación de los colaboradores se requiere el respaldo de una infraestructura capaz de identificar a las personas y de asociarlas a una ubicación simbólica (e.g., oficina o cafetería). En el apéndice A, se incluyen los sistemas más relevantes para monitorear la ubicación de personas y dispositivos.

Premisa de diseño
Las personas y los dispositivos que se encuentran en la misma ubicación están sujetos a las mismas variables físicas.

Basándonos en la premisa anterior proponemos el elemento **perceptor**, uno por cada ubicación considerada en el entorno colaborativo. Su función es la de monitorear las variables físicas presentes en la ubicación y determinar cuándo y qué eventos contextuales deben producirse. Estos eventos contextuales posteriormente se notifican

a un servidor contextual (denominado VIGÍA), encargado de decidir si hay un cambio en el contexto (cf. subsección 3.2.3).

A manera de ejemplo, un perceptor en la ubicación U podría observar algunas variables físicas como el estado de una impresora, el nivel de ruido, el nivel de luz, el número de personas en una habitación, la temperatura ambiente y el estado de un despliegue compartido táctil (cf. figura 3.8), pudiendo producir los siguientes eventos contextuales:

- La impresora I está fuera de servicio.
- El nivel de ruido es elevado en la sala de juntas S.
- El nivel de luz es muy bajo en la ubicación U.
- El colaborador C ha salido de la oficina.
- Una persona externa ha entrado a la oficina.
- La temperatura ambiente en el auditorio A: frío.
- El colaborador C ha comenzado a usar el despliegue táctil D.

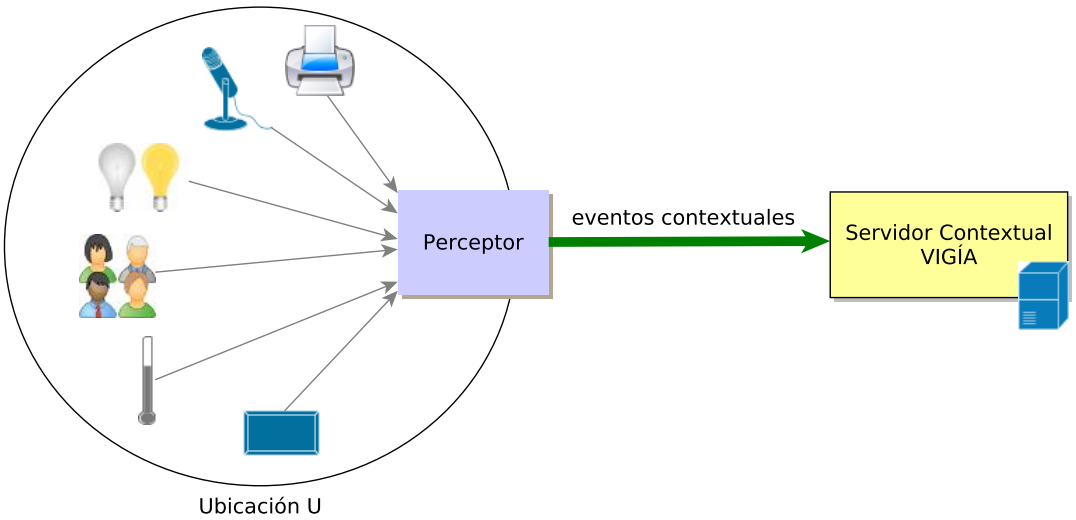


Figura 3.8: Elemento "perceptor"

No todos los cambios en las variables físicas producen un evento contextual. Un perceptor puede estar monitoreando un termómetro, sin embargo el aumento o disminución en un grado centígrado de la temperatura no produce por sí solo un evento contextual. En cambio, el perceptor debe comparar la temperatura ambiental con la definición de los posibles estados (frío, templado o cálido), de modo que solo si hay

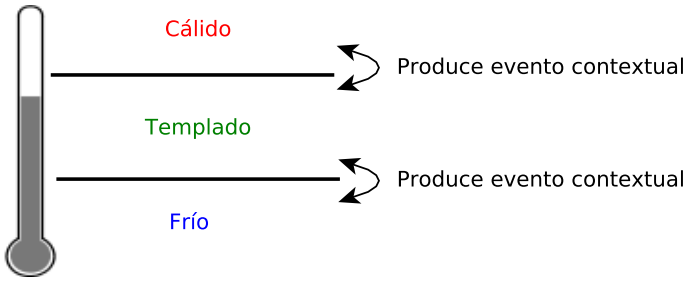


Figura 3.9: solo al cambiar entre frío, templado o cálido se produce el evento contextual correspondiente

un cambio en el estado actual se debe producir el evento contextual correspondiente (cf. figura 3.9).

La definición de un vocabulario contextual es esencial en el momento de establecer las condiciones para producir los eventos contextuales. Un vocabulario extenso permite hacer frente a descripciones abstractas en lugar de a datos técnicos [Baldauf et al., 2007], e.g., es más fácil entender el término "templado" que a su equivalente expresado como un intervalo de temperatura de 19 ^{o}C a 25 ^{o}C. En la tabla 3.2 se muestran algunos ejemplos de vocabulario contextual.

Variable física	Vocabulario contextual
Luz	Obscuro, normal, brillante
Temperatura	Frío, templado, cálido
Estado del despliegue táctil	Encendido, apagado
Número de personas	Habitación vacía, solo un usuario, habitación saturada

Tabla 3.2: Sensores de hardware comúnmente usados

Los perceptores también deben considerar pre-procesamientos, tales como la fusión de sensores y la calidad del contexto:

1. **Fusión de sensores:** se presenta cuando se usan dos o más sensores para medir la misma variable física. Por ejemplo, el número de personas en un cuarto puede ser obtenido a partir de varias técnicas, entre ellas procesamiento de video o procesamiento de audio. Mediante la fusión de sensores es posible obtener el número de personas con un factor de confianza mejor que el proporcionado por el video o el audio por separado [Calvary et al., 2001].

2. **Calidad del contexto:** La calidad en el contexto consiste en medir la imperfección de la información sensada, resultado del ruido, de fallas en los sensores y

de interrupciones en las redes de comunicación [McKeever et al., 2009]. El objetivo es considerar el error para tratar de corregirlo o para tomarlo en cuenta en etapas como la fusión de sensores.

Eventos contextuales lógicos

Reflejan los cambios en la dimensión interna de los entornos colaborativos: cambios en el estado de los proyectos y de los colaboradores, políticas organizacionales, reglas, procedimientos, actividades en progreso, roles de usuario, etc.

Estos eventos se originan principalmente en aplicaciones de software, bases de datos, repositorios de información y otros sistemas en donde se producen cambios lógicos. De esta manera, la fuente del evento contextual "Carlos se ha incorporado al equipo de trabajo" podría ser un sistema de administración de recursos humanos (SARH), en el que los datos del nuevo usuario han sido registrados.

En la tabla 3.3 se muestran algunos eventos contextuales lógicos y los respectivos sistemas de información donde pueden producirse.

Sistema de administración de recursos humanos (SARH)
Incorporación de un nuevo colaborador
Actualización de las habilidades de un colaborador
Actualización de la antigüedad de un colaborador
Inicio de las vacaciones de un colaborador
Repositorio de reglas y políticas de la organización
Inicia/termina intervalo de hora de comer
Inicia/termina intervalo de hora de entrada
Inicia/termina intervalo de hora de salida
Inclusión de una nueva política
Sistema de administración de proyectos
Cambio en la prioridad de un proyecto
Fecha límite de entrega alcanzada
Ocurrencia de un nuevo incidente
Asignación de un colaborador a una actividad

Tabla 3.3: Ejemplos de eventos contextuales lógicos

Generalmente, ya existen en las organizaciones los sistemas de software que conforman las fuentes de los eventos lógicos. Alguna de esa información necesita ser procesada mediante otras aplicaciones o módulos de software, e.g., un módulo manejador de alarmas de tiempo podría ser el responsable de que a las 6:00 de la tarde se dispare el evento "Hora de salida".

3.2.3 Detección de situaciones

Los eventos contextuales conducen al cambio de estado de las situaciones, i.e., conducen a su activación o desactivación. Por ejemplo, el evento contextual "Se ha actualizado la fecha de entrega del proyecto" puede provocar la activación de la situación "Proyecto en ruta crítica".

Una situación puede ser descrita a través de las siguientes propiedades:

- **Nombre:** identificador legible de la situación que nos permite distinguirla de otras.
- **Entidad relacionada:** se refiere a la entidad sobre la que recae directamente el estado de la situación, e.g., un usuario, un proyecto, una ubicación, etc.
- **Observadores:** consiste en las entidades que están interesadas en los cambios de estado de la situación.
- **Eventos contextuales:** eventos contextuales lógicos y físicos que pueden conducir al cambio de estado de la situación.
- **Condiciones:** requerimientos que deben ser verificados para determinar si la situación se activa o se desactiva.

Hemos clasificado a las situaciones en dos categorías: transitivas y permanentes.

Las situaciones **transitivas** son aquellas que no siempre están presentes y solo son importantes en el momento en que se activan. Para determinar si se activan es necesario evaluar las condiciones, aunque en muchos casos el conjunto de condiciones que se tienen que evaluar esté vacío, i.e., el evento contextual por sí solo activa la situación. Las situaciones transitivas no mantienen un estado persistente (activo/inactivo) de modo que su entidad relacionada es irrelevante (cf. figura 3.10).

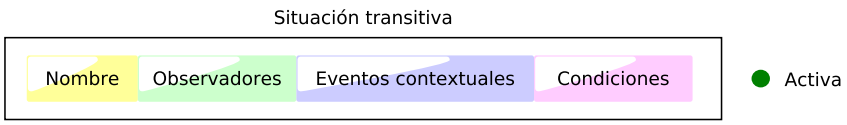


Figura 3.10: Propiedades de una situación transitiva

Por otro lado, las situaciones **permanentes** siempre están presentes, ya sea como activas o como inactivas, dependiendo de las condiciones que se cumplan y de los eventos contextuales que se generen (cf. figura 3.11).

Un ejemplo de situación permanente es la situación "Reunión" (cf. tabla B.7), la cuál es de interés para los colaboradores que se encuentra co-localizados en el lugar en donde se lleva a cabo la reunión. Las condiciones que se deben cumplir para activar esta situación son:

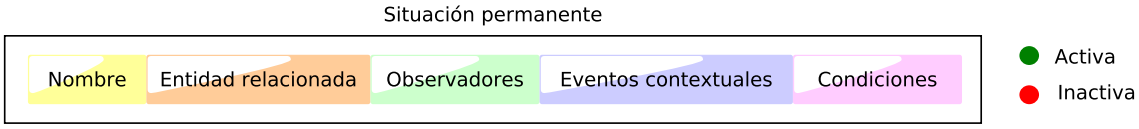


Figura 3.11: Propiedades de una situación permanente

1. El número de personas co-localizadas debe ser mayor a uno

2. Un proyector debe estar encendido

3. Todas las personas co-localizadas deben estar trabajando en el mismo proyecto

Mantener el estado de esta situación es importante para responder algunas preguntas (peticiones) que el sistema colaborativo pueda hacer en determinado momento, por ejemplo, ¿Se está llevando a cabo una reunión en la oficina de José?, ¿En qué lugar se está llevando a cabo una reunión?, etc.

Un ejemplo de una situación transitiva es la situación "Cambio de localización" (cf. tabla B.1). Esta situación puede ser explotada en mecanismos de conciencia grupal para indicar la localización de cada usuario en todo momento, por lo que solo es importante en el momento en que la situación se produce para reflejar el cambio de ubicación en la GUI.

En el apéndice B se presenta el modelado de las situaciones presentadas en las secciones 3.1.1 y 3.1.2 con base en las propiedades y clasificación propuestas.

Escenario

Mediante el siguiente escenario se pretende ilustrar la forma en la que se lleva a cabo la etapa de detección de situaciones de la arquitectura propuesta.

José entra a la sala de juntas y enciende el proyector, produciéndose los eventos contextuales: 1) "Colaborador {José} entra a la ubicación {sala de juntas}" y 2) "Un proyector se enciende". La situación "Reunión" no se activa con ninguno de los dos eventos, ya que las condiciones requeridas para su activación no se cumplen. Un minuto después entra Marco, generando el evento contextual "Colaborador {Marco} entra a la ubicación {sala de juntas}". Las primeras dos condiciones se cumplen, solo resta evaluar si ambos colaboradores están trabajando en el mismo proyecto. Sí es el caso, la situación "Reunión" se activa (cf. figura 3.12). Después de media hora, José sale de la habitación, produciéndose el evento contextual "Colaborador {José} sale de la ubicación {sala de juntas}". La primera condición deja de cumplirse, en consecuencia la situación "Reunión" se desactiva (cf. figura 3.13).

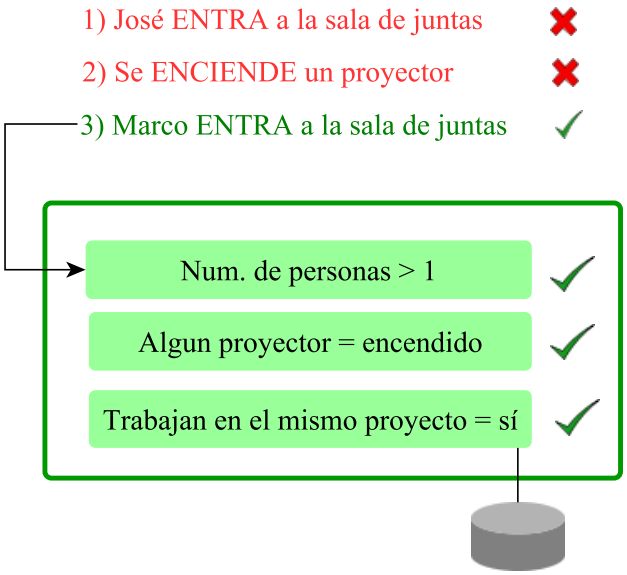


Figura 3.12: Situación "Reunión" activada

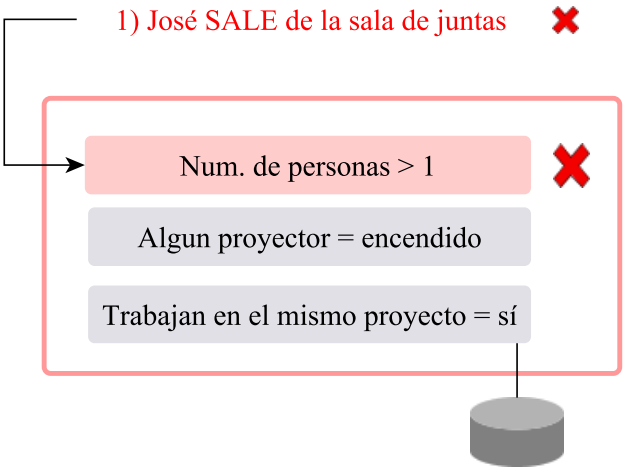


Figura 3.13: Situación "Reunión" desactivada

Procedimiento

Los eventos contextuales producidos se comunican a un servidor contextual, VIGÍA, quien se encarga de detectar si alguna nueva situación se activa o bien, se desactiva alguna situación presente (cf. figura 3.14).

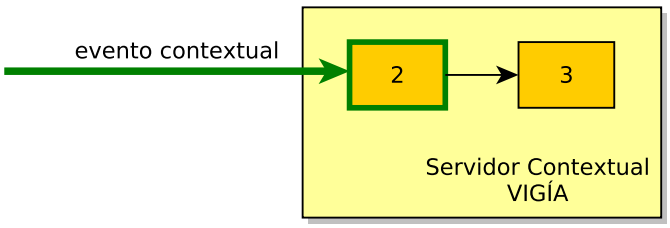


Figura 3.14: El servidor contextual VIGÍA es responsable de la fase de detección de situaciones

Los pasos para la etapa de detección de situaciones de la arquitectura contextual se muestra en el procedimiento 3.1. Básicamente consiste en lo siguiente:

Por cada evento contextual detectado se revisa cada una de las situaciones asociadas: si la situación es transitiva se evalúan sus condiciones, en caso de tener, y sí se activa, se notifica a sus observadores; mientras que si es permanente, se evalúan las condiciones de la situación para determinar su estado (activo/inactivo). La situación y su estado se notifica posteriormente a los observadores interesados (cf. sección 3.2.4).

Procedimiento 3.1 Procedimiento para llevar a cabo la detección de situaciones

Entrada: evento contextual e

1: Sea S el conjunto de situaciones asociadas a e.
2: **for** cada situación s en S **do**
3: **if** s es una situación transitiva **then**
4: activar ← evaluación de las condiciones de la situación s (en caso de no haber condiciones, evalúa como verdadero)
5: **if** activar es verdadero **then**
6: notificar s a sus observadores
7: **else**
8: /* s es una situación permanente */
9: evaluar las condiciones de la situación s y determinar su estado (activo/inactivo)
10: notificar s y su estado (activo/inactivo) a sus observadores

3.2.4 Comunicación de situaciones

Una vez que se ha detectado un cambio de estado de las situaciones se debe notificar a las entidades interesadas (observadores). El servidor contextual VIGÍA, el cual se encarga de la detección de cambios, también es responsable de notificarlos (cf. figura 3.15).

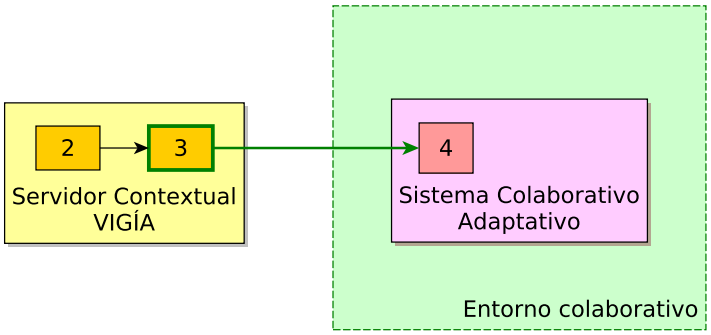


Figura 3.15: El servidor contextual VIGÍA es responsable de la fase de comunicación de situaciones.

Premisa de diseño
Los usuarios no están interesados en todos los cambios de estado de las situaciones, solo en aquellos que los afectan directamente.

No tiene sentido que se reciba notificación de todos los cambios contextuales producidos. Basados en esta premisa proponemos el uso del esquema de mensajes *Publish/Subscribe* [Eugster et al., 2003] para la comunicación de cambios en las situaciones. Este patrón consiste en una entidad generadora de mensajes (*publisher*) que no son enviados directamente a cada destinatario (*subscriber*), sino que simplemente se publican bajo cierto tópico sin el conocimiento de los posibles consumidores, i.e., es un sistema débilmente acoplado. A sí mismo los subscriptores expresan su interés en ciertos tópicos, sin el conocimiento de la entidad publicadora y posteriormente son notificados de los mensajes que coinciden con sus intereses (cf. figura 3.16).

De acuerdo a la arquitectura propuesta, cada nodo del sistema colaborativo tiene que subscribirse a los tópicos contextuales que son de su interés, e.g., un colaborador puede estar interesado en las situaciones que lo afectan a él directamente, a la ubicación en la que se encuentra, a los proyecto en los que está colaborando y al entorno colaborativo en general (cf. figura 3.17).

Escenario

Suponga que José y sus colegas se encuentran trabajando en la sala de juntas, realizando algunas actividades relacionadas con el proyecto Ciencias Básicas SEP-Conacyt.

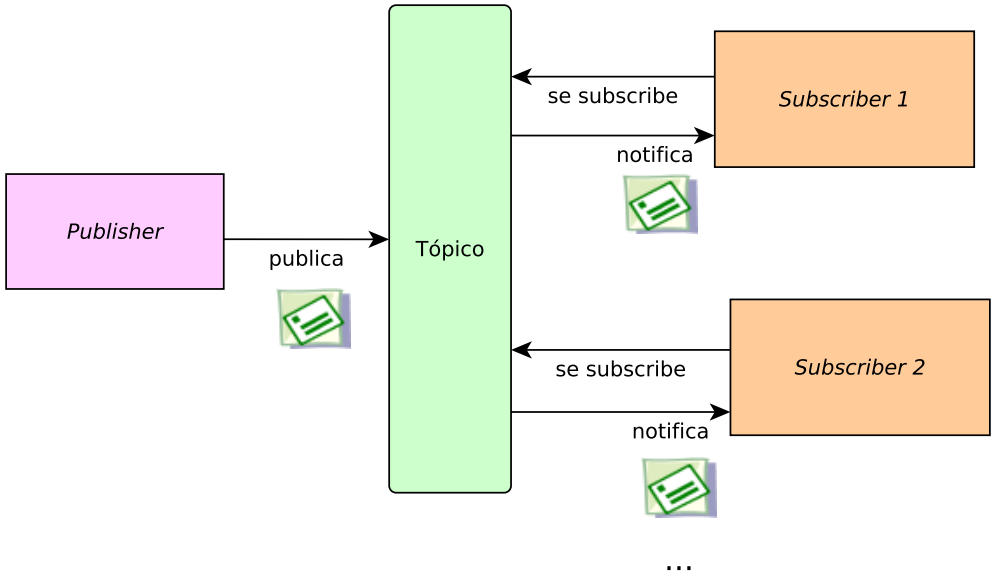


Figura 3.16: Esquema de mensajes *Publish/Subscribe*

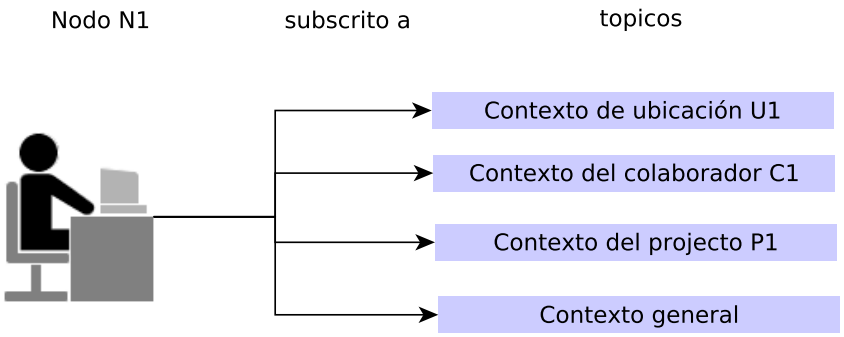


Figura 3.17: Un nodo del sistema colaborativo subscrito a algunos tópicos contextuales

José ingresa al sistema colaborativo a través de un dispositivo móvil previa validación de usuario. En el momento en que se le otorga acceso, el sistema se subscribe al contexto general del entorno colaborativo, al contexto del colaborador {José}, al contexto de la ubicación {Sala de juntas} y al contexto del proyecto {Ciencias Básicas SEP-Conacyt}.

Tiempo después José sale de la sala de juntas con su dispositivo móvil, en este momento, el sistema debe anular su subscripción al contexto de la ubicación {Sala de juntas} y cuando se detecte que José ha entrado a otra ubicación, e.g., la cafetería, se debe subscribir al contexto de la ubicación correspondiente.

Lo mismo aplica en el caso de que el colaborador sea reasignado a otro proyecto. Además, si el usuario trabaja simultáneamente en varios proyectos, el sistema tiene que suscribirse a todos ellos.

3.2.5 Adaptación del sistema colaborativo

Cuando se ha detectado y notificado un cambio en el contexto, el sistema colaborativo debe adaptarse convenientemente, para ello necesita de un módulo de adaptación.

Para realizar el proceso de adaptación en el sistema se hace uso del patrón de diseño **Observer** [Gamma et al., 1995], el cuál define una dependencia uno-a-muchos entre objetos, de manera que cuando un objeto (observable) cambia su estado, todos sus dependientes (observadores) son automáticamente notificados y actualizados en consecuencia.

En el caso de la arquitectura propuesta, el objeto observable corresponde al contexto (compuesto de las situaciones presentes y sus respectivos estados), mientras que uno de los observadores podría ser la GUI, la cuál debería adaptarse para reflejar los cambios contextuales (cf. figura 3.18).

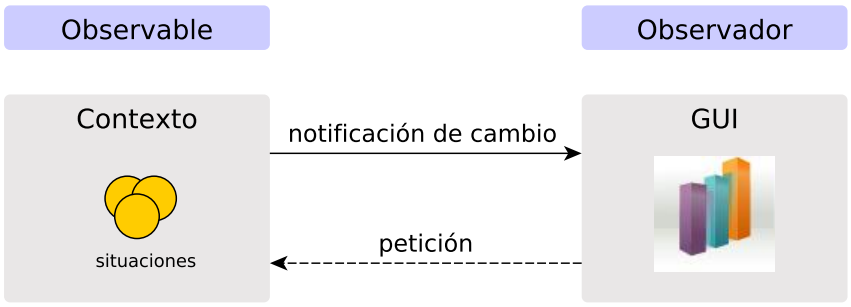


Figura 3.18: La fase de adaptación se basa en el patrón de diseño Observer

Dentro de la fase de adaptación también se deben considerar las **reglas de adaptación** que dictan las modificaciones que se deben hacer sobre el sistema colaborativo.

En su forma más simple estas reglas de adaptación son estáticas, i.e., siguen la forma **if** *situación detectada* **then** *adaptación subsecuente* (cf. figura 3.19).

1: **if** Reunión detectada **then**
2: desplegar mensaje "Se ha detectado una reunión"
3: buscar los documentos relevantes para la reunión y destacarlos

Figura 3.19: Las reglas de adaptación son estáticas (if-then)

Algunos ejemplos de adaptación se expusieron anteriormente en la sección 3.1, sin embargo la forma en que se debe adaptar el sistema colaborativo queda determinada por el dominio específico de aplicación.

3.2.6 Integración con un sistema colaborativo

Los componentes básicos de un entorno colaborativo típico son usuarios, dispositivos, recursos compartidos y un sistema colaborativo para coordinar esfuerzos (cf. figura 3.20). Los usuarios no necesariamente se encuentran en el mismo lugar, sino que pueden encontrarse en diferentes ubicaciones; incluso algunos pueden estar trabajando en el exterior a través de dispositivos móviles.

En cada ubicación es posible identificar dispositivos fijos (e.g., impresoras, proyectores y despliegues compartidos), cuya ubicación no cambia con frecuencia, así como dispositivos móviles (e.g., laptops, palms, celulares y tablets), que pueden ser fácilmente transportados por los usuarios.

Un sistema colaborativo se encarga de dar el soporte necesario para que los miembros del grupo de trabajo interactúen adecuadamente, proporcionando mecanismos de comunicación, resolución de conflictos y coordinación, entre otros.

La figura 3.21 muestra la integración de la arquitectura propuesta con un entorno colaborativo típico compuesto de usuarios, dispositivos y recursos compartidos (representados con letras mayúsculas), ubicaciones (representadas con círculos) y un sistema colaborativo (representado con una nube). El mapeo de estos elementos con las cuatro fases de la arquitectura contextual se describe a continuación:

1. el cuadro blanco en cada ubicación corresponde al perceptor que monitorea los sensores de hardware y determina los eventos contextuales físicos que deben producirse. El cuadro con relleno sólido representa las aplicaciones de software, bases de datos, repositorios de información y otros sistemas donde los eventos contextuales lógicos son producidos.

2. los eventos contextuales lógicos y físicos se notifican a un servidor contextual (VIGÍA) el cual lleva a cabo la fase de detección de situaciones, determinando qué situaciones se activan o desactivan.

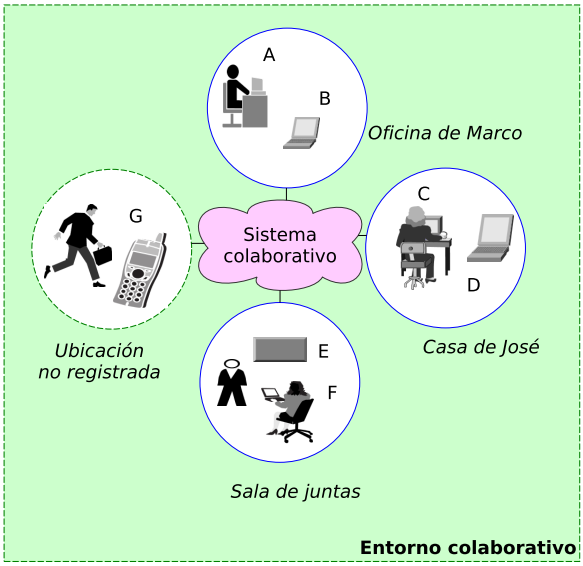


Figura 3.20: Entorno colaborativo típico: usuarios, dispositivos y sistema colaborativo

3. la fase de comunicación de cambios de estado de las situaciones también es realizada por el servidor contextual VIGÍA a través del esquema de mensajes *Publish/Subscribe*, con VIGÍA jugando el rol de entidad publicadora (*publisher*) y cada nodo del sistema colaborativo actuando como un subscriptor interesado en algunos tópicos contextuales (*subscribers*).

4. cada nodo del sistema colaborativo debe tener un módulo de adaptación para cambiar de acuerdo a las variaciones detectadas en las situaciones (contexto).

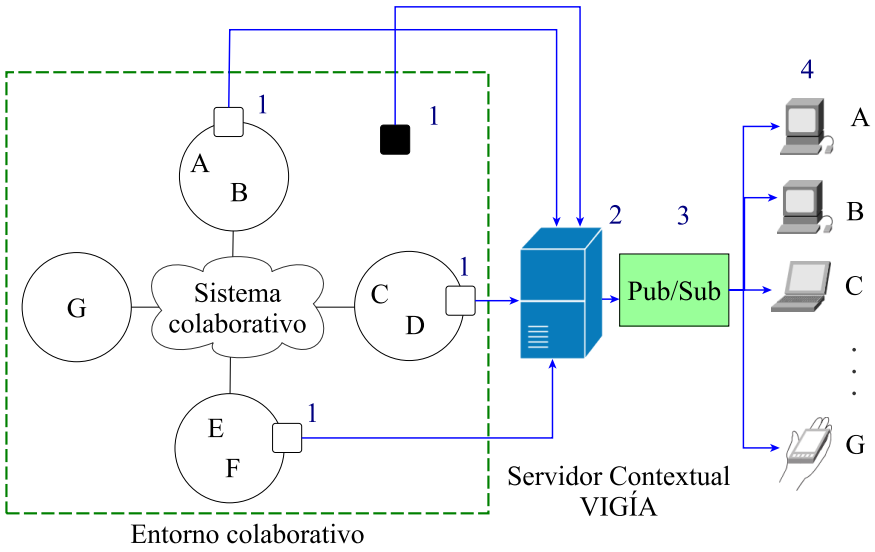


Figura 3.21: Integración de la arquitectura contextual propuesta con un entorno colaborativo típico

Capítulo 4

Implementación y validación

En este capítulo se presenta una implementación basada en el diseño de la arquitectura contextual propuesta, así como la descripción de una aplicación de prueba desarrollada. La sección 4 describe detalles de la implementación considerando aspectos como el lenguaje de programación, estructuras de datos, módulos, herramientas y tecnologías, sin embargo estos aspectos pueden ser reemplazados por los más adecuados dependiendo de las necesidades específicas del sistema colaborativo sobre el que se implemente la arquitectura. La sección 4.2 describe los detalles de la aplicación de prueba desarrollada, un editor colaborativo de mapas mentales, como medio de validación de la arquitectura.

4.1 Una implementación de la arquitectura

Después de definir las directivas generales que se deben seguir para desarrollar un sistema colaborativo contextual con base en la arquitectura propuesta (cf. sección 3.2), el siguiente paso es demostrar su viabilidad. Cada una de las fases de la arquitectura contextual puede ser implementada de distintas formas (e.g., empleando lenguajes de programación variados, haciendo uso de tecnologías existentes, o bien desarrollando cada elemento desde cero). En esta sección se describe una implementación desarrollada siguiendo las pautas establecidas en el diseño de la arquitectura contextual. Después de enumerar y describir brevemente los módulos de la implementación de la arquitectura contextual (cf. subsección 4.1.1), se exponen algunas decisiones generales consideradas en la implementación (cf. subsección 4.1.2). Finalmente se proporcionan detalles de cada módulo (cf. subsección 4.1.3 - 4.1.6).

4.1.1 Módulos de la implementación

La figura 4.1 muestra los componentes lógicos que integran la implementación de la arquitectura:

- **Perceptor:** módulo que corresponde a un simulador de eventos contextuales tanto físicos como lógicos, los cuales son notificados al servidor contextual VIGÍA.
- **VIGÍA:** módulo encargado de procesar los eventos contextuales para determinar si una situación se activa o desactiva (detección de situaciones). También notifica estos cambios a los interesados mediante el esquema de mensajes *Publish/Subscribe* (comunicación de situaciones).
- **Sistema colaborativo:** módulo que corresponde al software colaborativo (*groupware*) que proporciona los mecanismos y herramientas para que múltiples usuarios trabajen sobre la misma meta de manera simultánea. Debe adaptarse al recibir notificaciones de cambios en las situaciones.
- **SuperServer:** módulo que tiene acceso a la base de datos, por lo que es capaz de responder las peticiones sobre las entidades del entorno colaborativo: usuarios, proyectos, ubicaciones, dispositivos, etc.

4.1.2 Decisiones generales

La arquitectura contextual propuesta es independiente de los detalles de implementación, entre ellos el lenguaje de programación. En la implementación descrita en esta sección se usó Java, debido a que junto con C y C++ se encuentra dentro de los tres primeros lugares de popularidad entre desarrolladores como se muestra en la tabla 4.1 [TIOBE-Software, 2011].

Existen varios entornos de desarrollo (IDE) compatibles con Java, sin embargo se usó NetBeans IDE, porque está escrito en Java, siendo compatible con cualquier plataforma con JVM instalada, incluyendo Windows, Mac OS, Linux y Solaris.

4.1.3 Perceptor

Los eventos contextuales (cf. sección 3.2.2) se producen a partir del monitoreo de sensores físicos y sistemas de información. Sin embargo, el presente trabajo no se limita a eventos contextuales específicos, sino lo que se pretende es que su alcance sea más general. Un módulo de simulación de eventos contextuales es suficiente para el desarrollo de este trabajo de tesis, tomando en cuenta que la arquitectura considera

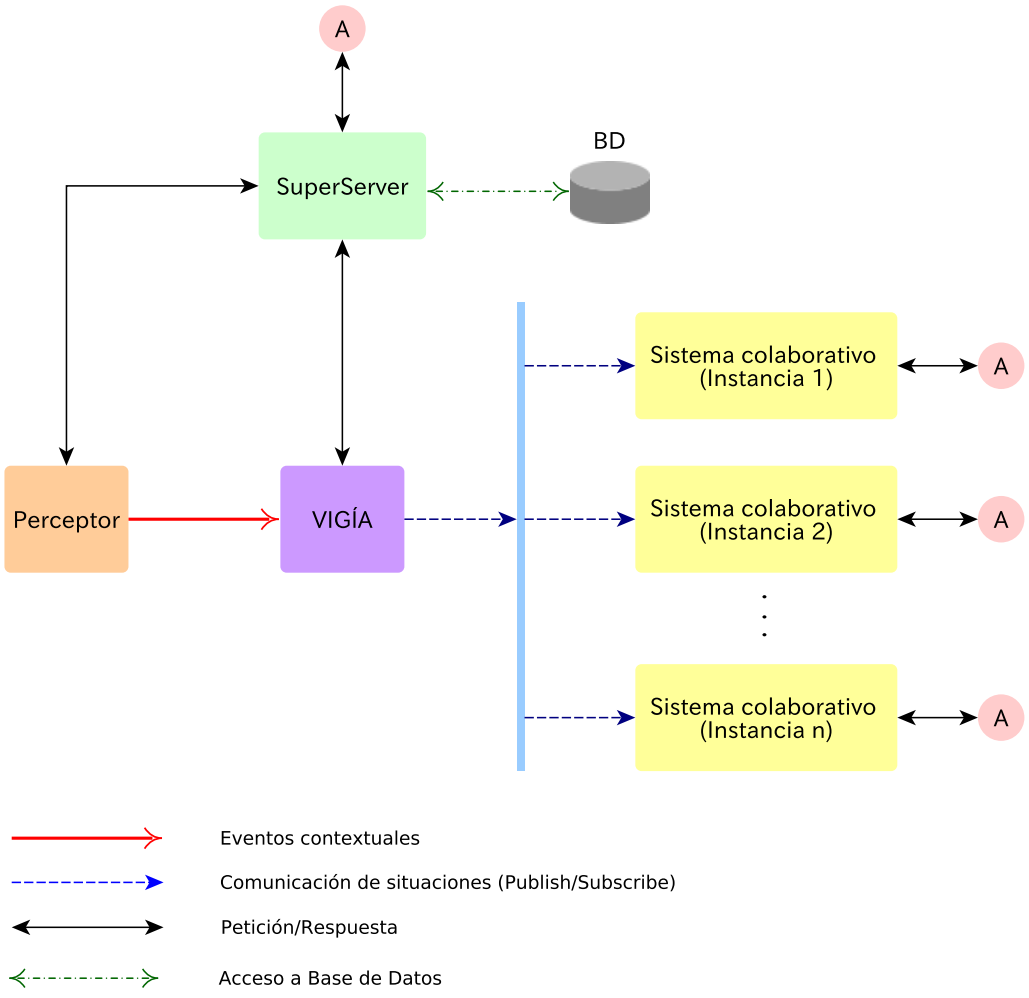


Figura 4.1: Componentes lógicos en la implementación de la arquitectura contextual

Lenguaje	Popularidad	Posición 2011	Posición 2010
Java	17.874 %	1	1
C	17.322 %	2	2
C++	8.084 %	3	3
C#	7.319 %	4	5
PHP	6.096 %	5	4
Objective-C	5.983 %	6	8
Visual Basic	5.041 %	7	7
Python	3.617 %	8	6
JavaScript	2.565 %	9	11
Perl	2.078 %	10	9
Ruby	1.502 %	11	10

Tabla 4.1: Índice TIOBE para noviembre de 2011 y 2010

como base a los eventos contextuales, independientemente del mecanismo a través del cual fueron generados.

Los eventos contextuales físicos y lógicos se activan de forma manual a través de una aplicación con interfaz gráfica de usuario. La figura 4.2 muestra una vista de dicha aplicación correspondiente a la activación de los eventos contextuales "Colaborador C entra a la ubicación U" y "Colaborador C sale de la ubicación U".

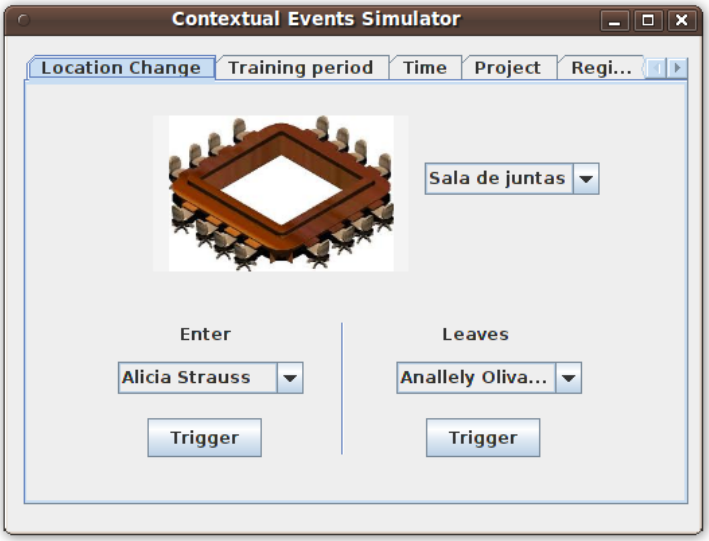


Figura 4.2: Módulo perceptor: un simulador de eventos contextuales

Los eventos contextuales soportados por el módulo simulador son los siguientes:

- Colaborador entra a una ubicación.

- Colaborador sale de una ubicación.
- Periodo de introducción activado.
- Periodo de introducción desactivado.
- Hora de comer inicia.
- Hora de comer termina.
- Intervalo de entrada inicia.
- Intervalo de entrada termina.
- Intervalo de salida inicia.
- Intervalo de salida termina.
- Estado de un proyecto actualizado.
- Fecha de entrega de un proyecto actualizada.
- Alerta: proyecto en ruta crítica.
- Nuevo colaborador registrado.
- Alerta: colaborador ausente.
- Colaborador ingresa al sistema.
- Colaborador sale del sistema.
- Persona externa entra a una ubicación.
- Persona externa sale de una ubicación.
- Proyector se enciende.
- Proyector se apaga.

Notificación de eventos contextuales

Las propiedades de un evento contextual son **nombre** y **parámetros**. El nombre corresponde al identificador del evento contextual, en tanto que los parámetros proporcionan información específica sobre el evento (cf. figura 4.3).

El número y tipo de parámetros es dependiente del evento contextual, por lo que para su representación se usó una estructura de datos dinámica: un mapa, donde la llave corresponde a un tipo de parámetro y el valor a una instancia específica de ese parámetro (cf. figura 4.4). Los eventos contextuales producidos se notifican al servidor

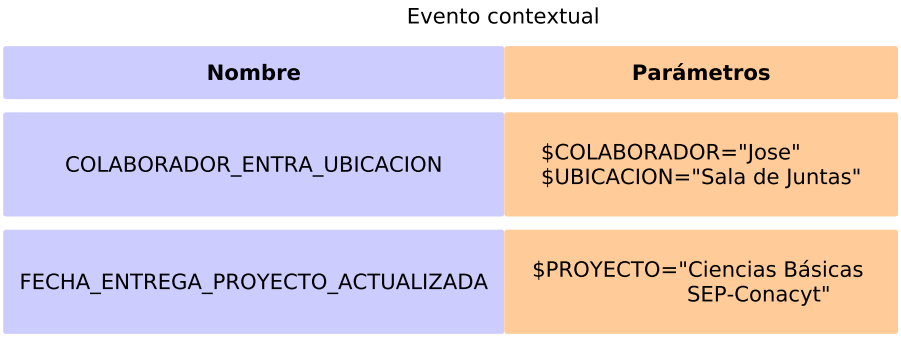


Figura 4.3: Propiedades de un evento contextual con sus respectivos ejemplos

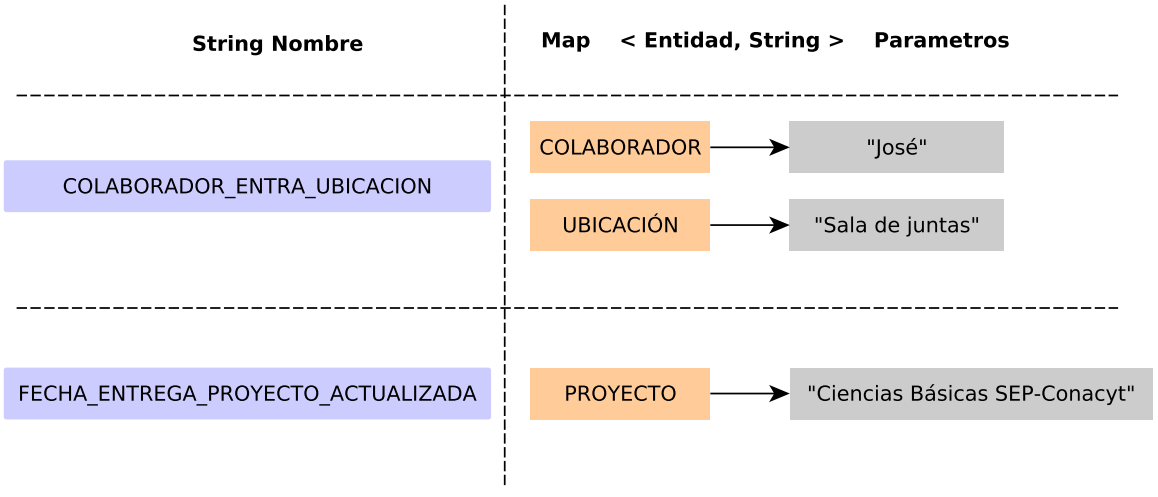


Figura 4.4: Mediante un mapa se representan los parámetros de los eventos contextuales

contextual VIGÍA, por medio de *sockets*, usando el método `writeObject` de la clase `ObjectOutputStream` [Oracle, 2011a], que lleva a cabo la transmisión de objetos a través de la serialización de objetos.

4.1.4 VIGÍA

Este módulo se encarga de las fases de detección y comunicación de situaciones, para lo cual debe recibir los eventos contextuales, procesarlos para obtener los cambios de contexto y finalmente notificar estos cambios. Los detalles de estos pasos se describen a continuación.

Recepción de eventos contextuales

El servidor contextual VIGÍA debe estar listo para escuchar los eventos contextuales producidos por diferentes fuentes (cf. figura 4.5). En la implementación desarrollada se usaron *sockets* para la comunicación de eventos contextuales y para soportar el manejo de peticiones provenientes de más de un cliente, el servidor contextual VIGÍA se implementó como un servidor multi-hilo, el cual crea un hilo para atender peticiones por cada cliente. Un hilo es una secuencia de instrucciones que se ejecutan de forma independiente del programa y de los otros hilos.

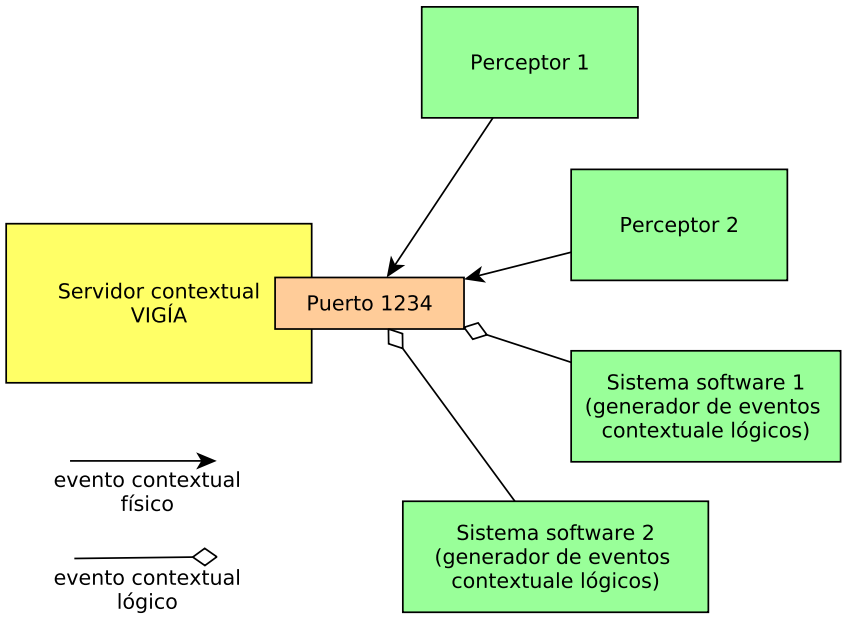


Figura 4.5: Servidor contextual VIGÍA escuchando eventos contextuales de diferentes fuentes

Una vez establecido un hilo de comunicación entre un cliente y el servidor, éste se bloquea en espera de un nuevo evento contextual a través del método `readObject` de la clase `ObjectInputStream` [Oracle, 2011b], el cual reconstruye el evento contextual mediante la técnica de serialización.

Procesamiento de eventos contextuales

Cada evento contextual que recibe VIGÍA debe ser procesado como se mostró previamente en el procedimiento 3.1. El primer paso consiste en identificar las situaciones que pueden verse afectadas por el evento contextual que está siendo procesado. A continuación se describe formalmente la relación entre los eventos contextuales y las situaciones asociadas a ellos.

Sea $S = \{S_1, S_2, ..., S_n\}$ el conjunto de situaciones soportadas por el sistema colaborativo y sea $C = \{C_1, C_2, ..., C_m\}$ el conjunto de eventos contextuales que pueden ser producidos. El conjunto de situaciones asociadas a $C_x \in C$ está dado por:

$$asoc(C_x) = S_A \subset S$$

donde S_A está determinado por el dominio específico del sistema colaborativo, i.e., las relaciones se obtienen de la etapa de análisis de requerimientos, propia del desarrollo de software. Por ejemplo, en la figura 4.6 las situaciones asociadas a C_3 son:

$$asoc(C_3) = \{S_1, S_m\}$$

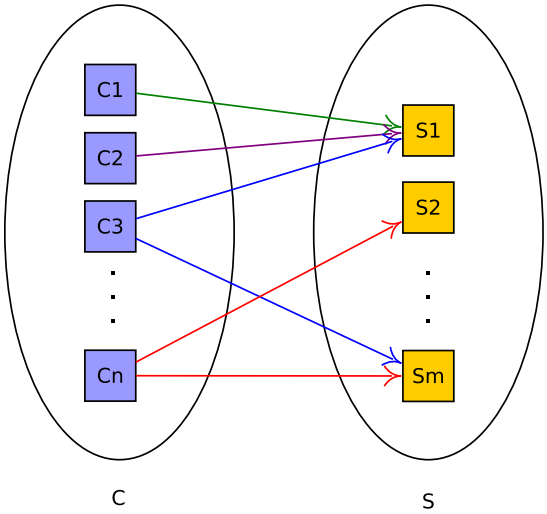


Figura 4.6: Relación de situaciones asociadas a un evento contextual

Posteriormente cada situación asociada se analiza de forma independiente. Si la situación es transitiva simplemente se notifica su activación, pero en el caso de que

sea permanente es necesario llevar a cabo los pasos de: 1) completitud de parámetros, 2) evaluación de condiciones, 3) actualización del contexto y 4) notificación de los cambios. Los primeros tres pasos se exponen enseguida, mientras que el cuarto se explica con detalle en la subsección inmediata.

Completitud de parámetros

Los parámetros de los eventos contextuales contienen información útil en las etapas de detección y comunicación de situaciones.

Analicemos el caso del evento contextual "Fecha de entrega del proyecto actualizada" y una de sus situaciones asociadas "Proyecto en ruta crítica":

1. Una condición que se debe evaluar para determinar el estado de la situación es que el número de días para entregar el proyecto sea menor o igual a diez
2. Posteriormente se debe notificar el estado de la situación a los observadores, que en este caso son los subscritos al tópico del proyecto en cuestión.

Estas dos acciones se mapean a métodos que necesitan como parámetro de entrada a un proyecto, el cual es proporcionado en este caso por el evento contextual (cf. figura 4.7).

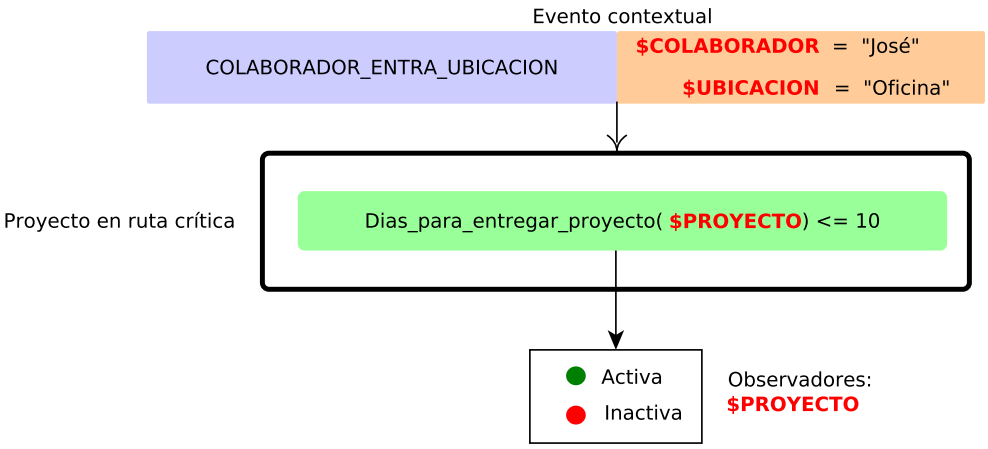


Figura 4.7: Algunos parámetros proporcionados por el evento contextual son usados en varios pasos del procesamiento de eventos contextuales

Sin embargo, en algunos casos, es posible que se requieran parámetros que no son proporcionados por el evento contextual detonante (cf. figura 4.8). En estos casos es necesario realizar una etapa previa para poder obtener la información necesaria a partir de los parámetros del evento contextual.

Si en el procesamiento de eventos contextuales se requiere un $PROYECTO, pero en los parámetros del evento contextual solo se proporciona un $COLABORADOR, es necesaria la etapa de completitud de parámetros, para obtener el $PROYECTO a partir del $COLABORADOR, e.g., a tráves del método `getProjectCollaboratorIsWorkingAt` para obtener el proyecto en el cual el colaborador está trabajando.

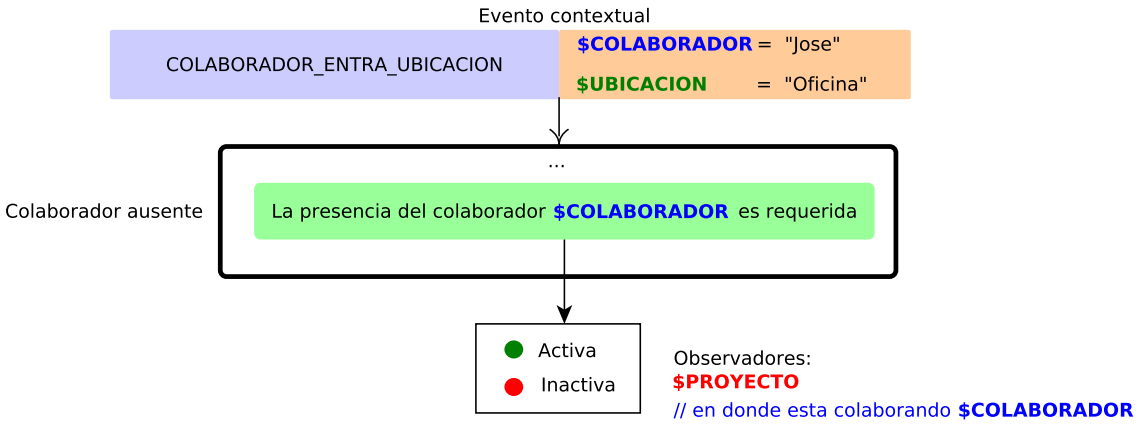


Figura 4.8: En algunos casos, es posible que se requieran de parámetros que no son proporcionados por el evento contextual

Evaluación de condiciones

Para conocer las condiciones de activación/desactivación de una situación se requiere de una etapa de análisis de requerimientos, lo que quizá corresponde a la mayor dificultad al momento de modelar una situación, sin embargo esta étapa está presente prácticamente en cualquier desarrollo de software. Teniendo identificadas las condiciones de una situación, éstas pueden ser programadas como métodos booleanos, por ejemplo:

- `workingOnSameProject( List<User> users )`
- `isProjectorOn( Projector p )`
- `countPeopleAtLocation( Location l ) > n`
- `getRemainingDaysToDeliverProject(p) > n`

Resulta intuitivo usar métodos booleanos y operadores lógicos para construir las condiciones de activación y desactivación de una situación. Otra ventaja de esta solución es que cada método puede ser reusado para definir nuevas situaciones de manera que, por medio de interfaces bien definidas, es posible alcanzar un alto nivel de formalidad [Strang and Linnhoff-Popien, 2004].

El procedimiento 4.1 ilustra una instancia de la evaluación de las condiciones para la situación "Reunión".

Procedimiento 4.1 Pseudocódigo para la evaluación de la situación "Reunión"

Entrada: evento contextual e

1: $l \leftarrow$ valor del parámetro $LOCATION en e
2: **if** el número de personas co-localizadas en l es mayor a uno **and** un proyector está encendido en l **and** las personas co-localizadas en l trabajan en el mismo proyecto **then**
3: estado $\leftarrow$ activado
4: **else**
5: estado $\leftarrow$ desactivado
6: **return** estado

Actualización del contexto

El contexto está conformado por las situaciones y su respectivo estado que afectan a las diferentes entidades en el entorno colaborativo. La estructura de datos que se usa para representar al contexto es un mapa, donde la llave corresponde a una entidad específica (e.g., proyecto 1, colaborador 5, etc.) y el valor corresponde a una lista con las situaciones que afectan dicha entidad, donde cada elemento de la lista corresponde a una 2−tupla de la forma <nombre_situacion, estado> (cf. figura 4.9).

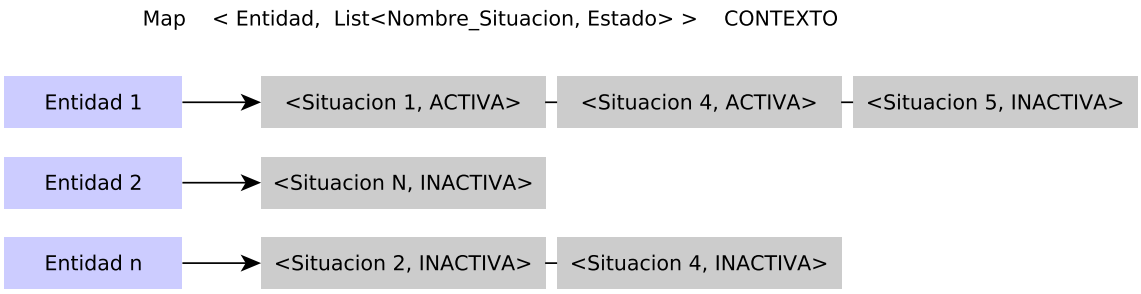


Figura 4.9: Estructura de datos usada para representar el contexto

A través de los métodos `put`, `get` y `add` de los mapas y listas en Java, se actualiza la estructura de datos para que se refleje el estado actual del contexto ante cada cambio detectado en las distintas situaciones.

Comunicación de situaciones

Los cambios en los estados de las situaciones se notifican a los interesados por medio del esquema de mensajes *Publish/Subscribe*.

Existen en la actualidad varias implementaciones del esquema de *Publish/Subscribe*, e.g., Apache ActiveMQ, OpenJMS, JBoss Messaging, RabbitMQ, WebSphere Application Server etc. La selección de la tecnología a usar depende de los requerimientos específicos del sistema colaborativo sobre el que se esté implementando la arquitectura contextual. Para fines de prueba se hizo uso de la plataforma RabbitMQ que provee distintos mecanismos de mensajes para aplicaciones, además de ser software libre y de contar con librerías para Java [VMware, 2011].

Los elementos considerados por RabbitMQ son los siguientes:

- *Productor* es una aplicación que envía mensajes.
- *Cola de mensajes* es un buffer que almacena mensajes.
- *Consumidor* es una aplicación que recibe mensajes.

La idea principal del modelo de mensajes de RabbitMQ es que el productor nunca envía los mensajes directamente a la cola, en cambio el productor envía los mensajes a una central (*exchange*). Esta central recibe como entrada los mensajes de los productores, los cuales son distribuidos a las colas de mensajes de acuerdo a las subscripciones de los consumidores a ciertos tópicos (cf. figura 4.10).

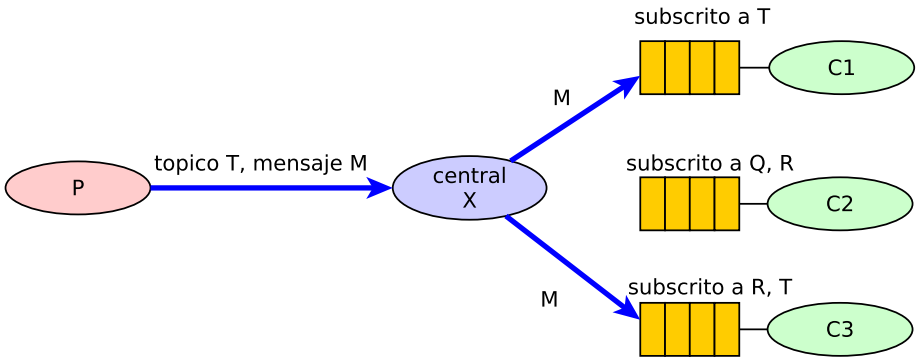


Figura 4.10: Publicación de mensajes en RabbitMQ

El procedimiento 4.2 muestra los pasos para la publicación de mensajes en RabbitMQ. Básicamente se establece un canal de comunicación con el servidor que distribuye los mensajes a través de una central. Una vez establecido el canal de comunicación se publican los mensajes (bajo cierto tópico) a través del método `basicPublish`.

Los tópicos contemplados en la implementación de la arquitectura contextual son los siguientes:

- **topico_general:** bajo este tópico se publican situaciones de interés para todas las entidades del entorno colaborativo.

Procedimiento 4.2 Pseudocódigo para la publicación de mensajes en RabbitMQ

Entrada: tópico T, mensaje M
1: Crear una conexión al servidor y un canal de comunicación
2: Declarar una central X
3: Publicar el mensaje mediante el método `basicPublish(`X, T, M`)`

- **topico_colaborador_idColaborador:** se publican las situaciones de interés para el colaborador con el *id* indicado.
- **topico_proyecto_idProyecto:** se publican las situaciones para los interesados en el proyecto con el *id* indicado, e.g., los usuarios que colaboran en dicho proyecto.
- **topico_ubicacion_idUbicacion:** se publican las situaciones para los interesados en la ubicación con el *id* indicado, e.g., los usuarios co-localizados en dicha ubicación.

Por otro lado, los mensajes son cadenas de texto que representan a las situaciones que necesitan ser comunicadas y que siguen el patrón:

$$NombreSituacion + ListaParametros + Estado$$

donde:

$+$: Indica la operación de concatenación.

NombreSituacion : Corresponde a la cadena con el nombre de la situación a ser comunicada

ListaParametros: Corresponde a la cadena conformada por la concatenación de los parámetros disponibles (aquellos proporcionados por el evento contextual que provocó la actualización de la situación, más los inferidos en la etapa de completitud de parámetros). Cada parámetro se representa con una cadena de la forma $\$parametro_N = valor_N$ con

- $N >= 0$,
- $parametro_i$ corresponde al identificador del parámetro i
- $valor_i$ corresponde al valor asociado al identificador del parámetro i

Por ejemplo, la cadena `$COLLABORATOR=5` es un ejemplo válido que indica que el parámetro `COLLABORATOR` tiene como valor a `5`, dicho número puede corresponder al identificador numérico de cierto colaborador.

Estado: corresponde a una cadena que indica el estado de la situación (activo o inactivo)

- es cadena vacía en caso de que la situación sea transitiva
- tiene la forma $\$estado = [activo|inactivo]$ en caso de que la situación sea permanente

Algunos ejemplos de instancias de los mensajes/situaciones comunicadas son:

- `REUNION$estado=inactivo` (situación permanente)
- `PERIODO_INTRODUCCION$COLLABORATOR=5$estado=activo` (situación permanente)
- `COLABORADOR_SALE_UBICACION$COLLABORATOR=2$LOCATION=1` (situación transitiva)

4.1.5 Sistema colaborativo

El sistema colaborativo debe suscribirse a los tópicos contextuales que le son de interés. El procedimiento 4.3 muestra los pasos para la recepción de mensajes/situaciones en RabbitMQ.

Procedimiento 4.3 Recepción de mensajes en RabbitMQ

1: Crear una conexión al servidor y un canal de comunicación
2: Declarar una central X (`exchangeDeclare`)
3: Declarar una cola de recepción de mensajes Q (`queueDeclare`)
4: Suscribir la cola Q a los tópicos contextuales de interés (`queueBind`)
5: **loop** {Se bloquea hasta recibir un nuevo mensaje y entonces se adapta}
6: $mensajeContextual \leftarrow$ `nextDelivery().getBody()`
7: ADAPTACION(*mensajeContextual*)

El mensaje contextual es recibido y posteriormente es enviado al módulo de adaptación del sistema colaborativo, en donde se identifica la situación y se aplican las reglas estáticas de adaptación correspondientes. Dichas reglas estáticas tienen la forma:

if *situación detectada* **then** *adaptación subsecuente*

La interfaz gráfica del sistema colaborativo se basa en el *toolkit Swing* de Java [Oracle, 2011c], el cual proporciona componentes como botones, etiquetas, controles para tablas, listas, árboles. etc. La librería de *Swing* hace uso del patrón de diseño MVC (Modelo/Vista/Controlador) en el cual se desacoplan los datos existentes (modelo) de la forma en que son visualizados por el usuario (vista). En tanto, el controlador actúa como elemento intermediario, recibiendo las entradas del usuario e

indicando al modelo y a la vista los cambios que deben hacer de acuerdo con esas entradas

La mayoría de los componentes de *Swing* tienen asociado un modelo que, al ser modificado de alguna manera, dispara eventos para que su representación gráfica se actualice acordemente (patrón de diseño *Observer*). Varias reglas de adaptación pueden implicar la modificación de los datos del modelo, lo que se traduce en una actualización inmediata de la forma en que se visualizan los componentes asociados.

4.1.6 SuperServer

Se trata del único módulo que tiene acceso a la base de datos, por lo que es capaz de responder las peticiones de consulta y actualización de otros módulos[1] sobre las entidades del entorno colaborativo: usuarios, proyectos, ubicaciones, dispositivos, etc.

La comunicación entre los distintos módulos y SuperServer se implementó a través de *sockets*, específicamente usando el *framework* de aplicaciones distribuidas Apache Mina [Foundation, 2011], el cual provee una API para comunicaciones asíncronas en aplicaciones de red.

En la implementación desarrollada se consideraron algunas entidades típicas de un entorno colaborativo: usuarios, proyectos, ubicaciones y dispositivos. Otras entidades más pueden ser consideradas según las necesidades particulares del sistema colaborativo contextual que se esté desarrollando. La figura 4.11 muestra el diagrama Entidad-Relación de las entidades consideradas; a partir de este diagrama se crearon las tablas correspondientes de la base de datos (cf. tabla 4.2).

Tabla	**Descripción**
collaborator	Información personal de cada usuario registrado en el entorno colaborativo.
location	Información sobre las ubicaciones consideradas dentro del entorno colaborativo.
device	Características de los dispositivos registrados.
project	Información sobre los proyectos.
collaborator_project	Asocia colaboradores con proyectos.
general_situation	Almacena las situaciones que afectan a todo el entorno colaborativo.
document	Almacenamiento de los documentos derivados de la aplicación colaborativa de prueba explicada en la sección 4.2.

Tabla 4.2: Descripción de las principales tablas de la base de datos

[1] Perceptores, servidor contextual VIGÍA e instancias del sistema colaborativo

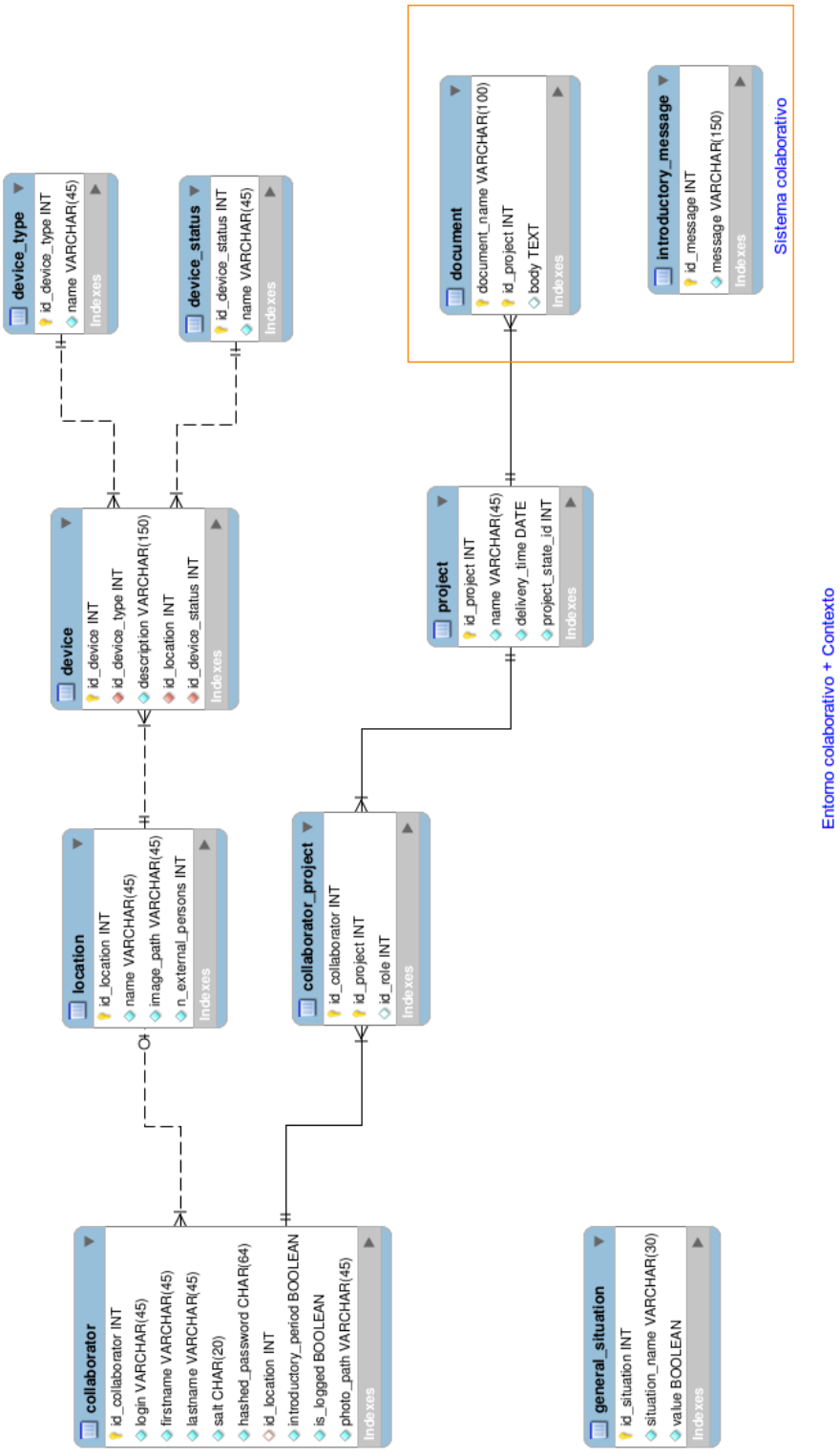


Figura 4.11: Modelo Entidad-Relación de las entidades del entorno colaborativo

La implementación del módulo SuperServer sigue una arquitectura de tres capas:

1. **comunicación:** conjunto de clases que escuchan las peticiones (*requests*), las decodifican y las mapean con los respectivos métodos locales que dan solución a sus peticiones. En esta capa se hizo uso de Apache Mina (basado en sockets) para el manejo de peticiones distribuidas.

2. **lógica:** conjunto de métodos locales que corresponden a la lógica de negocios de la aplicación.

3. **acceso a la base de datos:** conjunto de clases con métodos para realizar consultas a la base de datos (DAOS). En esta capa se usó Spring JDBCTemplate el cual simplifica el acceso a la base de datos a través de una API que soporta encapsulación de datos en objetos y manejo de excepciones.

El manejador de base de datos usado fue MySQL, por ser multi-plataforma, proveer un soporte completo a las funciones de gestión de bases de datos relacionales, además de ser multihilo, multiusuario y de código libre (bajo la licencia GNU GPL).

La figura 4.12 ilustra la arquitectura en capas del módulo SuperServer, indicando las tecnologías y manejador de bases de datos usados.

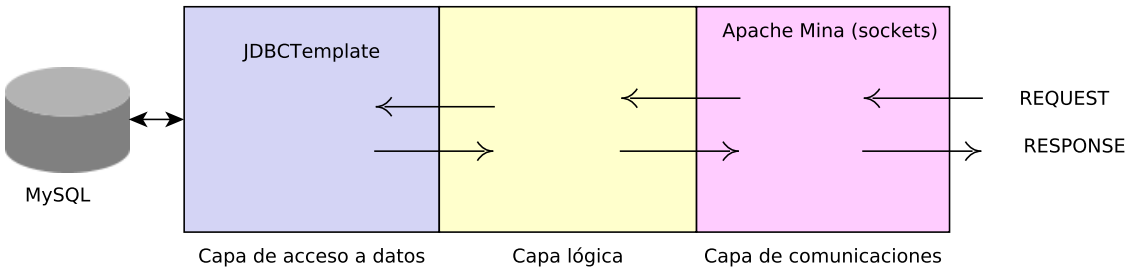


Figura 4.12: Arquitectura en capas del módulo SuperServer

4.2 Validación

En esta sección se describe un sistema colaborativo de prueba, desarrollado con el objetivo de validar el diseño de la arquitectura contextual propuesta. Primero se hace una descripción general de dicho sistema (cf. sección 4.2.1) y posteriormente se dan detalles de las capacidades adaptativas/contextuales del mismo (cf. sección 4.2.2).

4.2.1 Descripción general del sistema colaborativo

El sistema colaborativo de prueba se trata de un **editor colaborativo de mapas mentales**, cuya interfaz gráfica de usuario (GUI[2], por sus siglas en inglés) consta de dos áreas: el área de edición y el área de conciencia grupal (cf. figura 4.13).

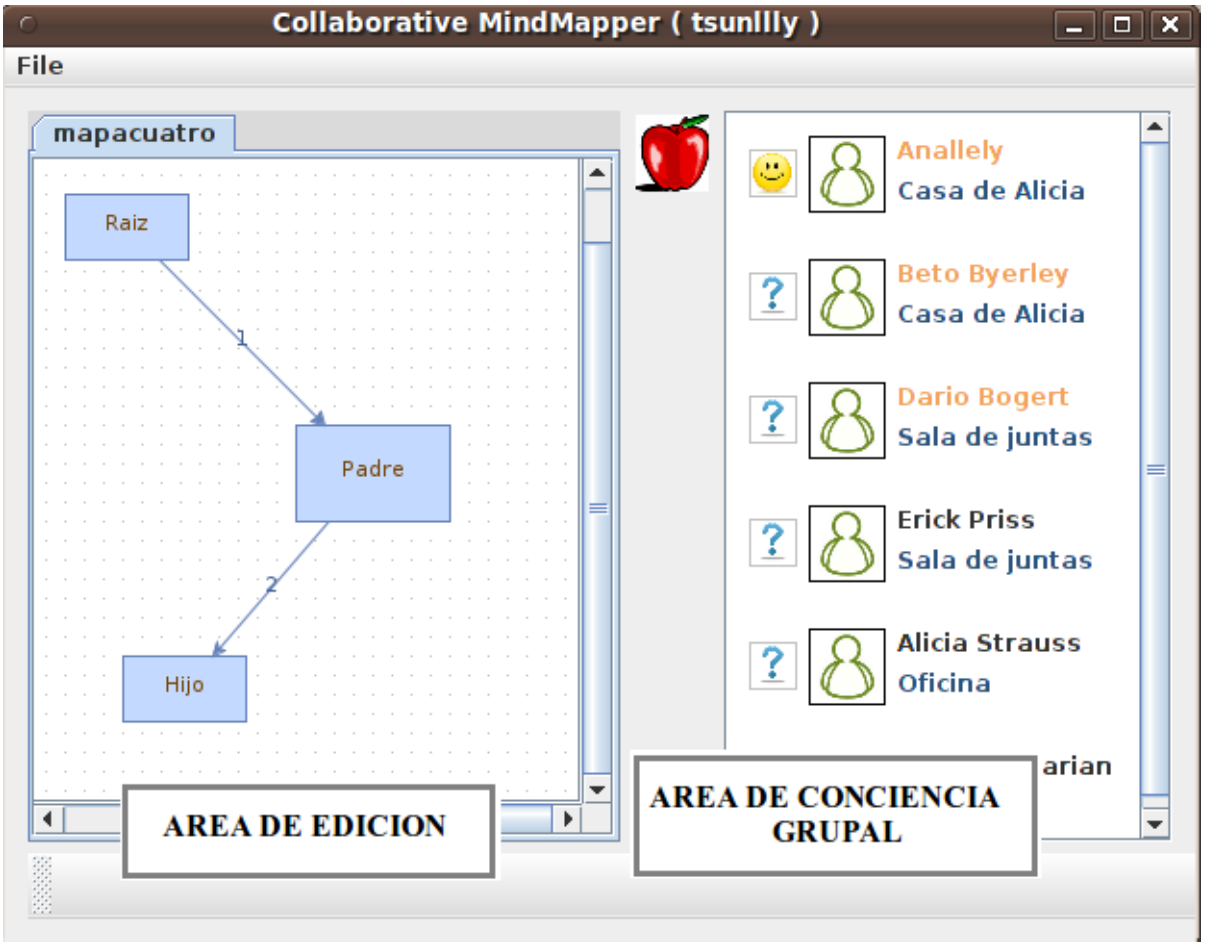


Figura 4.13: Interfaz gráfica de usuario del editor colaborativo de mapas mentales

Los módulos principales de este sistema colaborativo son: 1) ingreso al sistema, 2) edición de mapas, 3) conciencia de grupo y 4) adaptación. Los primeros tres módulos se explican a continuación, mientras que el último se detalla en la siguiente subsección.

Módulo de ingreso al sistema

Para ingresar al sistema se debe proporcionar el *login* de usuario y la contraseña, los cuales son validados para otorgar acceso al sistema colaborativo (cf. figura 4.14). Una vez otorgado el acceso, el sistema se subscribe a los siguientes tópicos contextuales:

[2]Graphic User Interface

1. tópico del entorno colaborativo general
2. tópico del usuario
3. tópico de la ubicación donde se encuentra el usuario

Figura 4.14: Diálogo de ingreso al sistema colaborativo

Módulo de edición de mapas

Proporciona los mecanismos de producción del sistema colaborativo, en este caso la edición de mapas mentales que incluye las siguientes herramientas:

- Agregar nodos nuevos.
- Agregar hijos a los nodos existentes.
- Editar texto de los nodos y conectores.
- Eliminar nodos y conectores.
- Cambiar el tamaño de los nodos.
- Cambiar la posición de los nodos.

En el desarrollo e implementación de este módulo, se hizo uso de la librería JGraphX [Benson and Alder, 2011] que proporciona varias funcionalidades para la interacción con diagramas y grafos.

El sistema soporta la existencia de diferentes documentos y que varios colaboradores trabajen simultáneamente sobre el mismo documento. Los cambios hechos por un usuario se reflejan inmediatamente en la pantalla de cada uno de los usuarios que trabajan en el mismo documento.

Un usuario puede elegir entre las opciones de crear un nuevo documento, trabajar sobre uno ya existente o dejar de trabajar en alguno.

Nuevo documento

Al seleccionar esta opción, se muestra al usuario un diálogo en el cual debe introducir el nombre del nuevo documento y el proyecto al cual estará asociado (cf. figura 4.15). En el momento en que el usuario confirma los datos ingresados, el documento se crea y se pone a disposición de todos los colaboradores, al mismo tiempo que el sistema colaborativo se subscribe al **tópico del proyecto** al cual está asociado el documento.

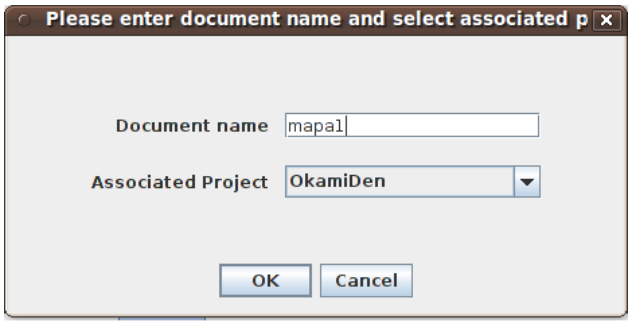


Figura 4.15: Diálogo para crear un nuevo documento

Abrir documento

Si el usuario selecciona esta opción, se le muestra un diálogo donde debe seleccionar el proyecto al que pertenece el documento que desea abrir y posteriormente el documento mismo (cf. figura 4.16). El sistema colaborativo solicita el documento compartido al servidor SuperServer y lo carga en la pantalla del usuario, al mismo tiempo que se subscribe al **tópico del proyecto** al cual está asociado el documento.

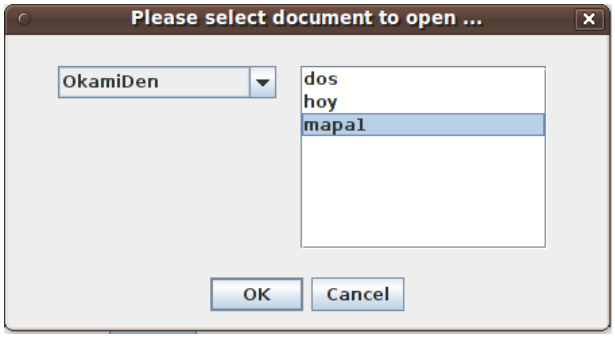


Figura 4.16: Diálogo para abrir un documento existente

Cerrar documento

Ante esta acción, el sistema colaborativo deja de recibir actualizaciones sobre el estado del documento y cancela su subscripción al tópico contextual del proyecto asociado con el documento que se cierra.

Módulo de conciencia grupal

En el lado izquierdo de la GUI del sistema colaborativo se encuentra el área de conciencia grupal, que corresponde a una **barra de situaciones** y una **barra de colaboradores** (cf. figura 4.17). En la barra de situaciones se muestran algunos íconos correspondientes a las situaciones activas (e.g., hora de comer, intervalo de inicio de labores, intervalo de hora de salida y reunión) y en la barra de colaboradores se muestra una lista de colaboradores, destacando:

- *Nombre* del colaborador;
- Nombre de la *ubicación* en donde el colaborador se encuentra (o un signo de interrogación en caso de no conocerse);
- Si el colaborador se encuentra en *periodo de introducción* o no (en el primer caso se indica con un ícono de signo de interrogación en la parte izquierda y en el segundo caso con un ícono de cara amarilla);
- Si el colaborador está *ausente* (en este caso se indica con un ícono de signo de admiración rojo en el lado izquierdo del nombre).

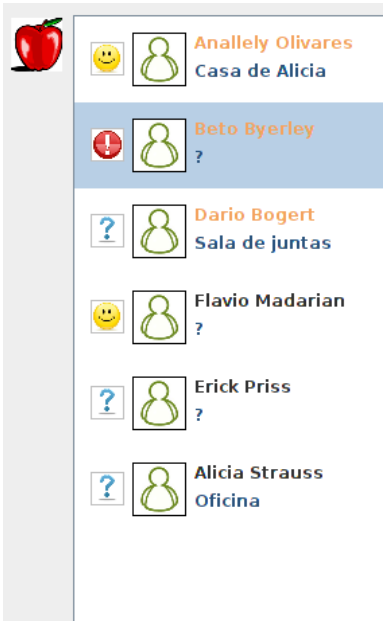


Figura 4.17: Área de conciencia grupal: barra de situaciones y barra de colaboradores

4.2.2 Módulo de adaptación

Las situaciones planteadas en la sección 3.1.1 fueron consideradas en el editor de mapas colaborativo, bajo el diseño e implementación de la arquitectura contextual propuesta. A continación se describen los detalles de algunas de estas situaciones.

Situación: "Colaborador cambia su ubicación"

El diagrama de secuencia mostrado en la figura 4.19 muestra el proceso de activación y adaptación de la situación "Colaborador cambia su ubicación". Los pasos involucrados en este proceso para cada módulo son los siguientes:

SuperServer

0. Iniciar SuperServer, pues es el módulo al que se le hacen peticiones sobre la base de datos.

Perceptor

1. Iniciar Perceptor, para lo cual se hacen peticiones a SuperServer para obtener la información necesaria de la base de datos.
2. En la pestaña Location Change de la GUI (cf. figura 4.2) se puede simular que un usuario entra o sale de una ubicación, e.g., Beto Byerley entra a la oficina.
   - 2.a Se notifica a SuperServer para que cambie la ubicación en la base de datos.
   - 2.b Se produce el evento contextual, e.g., `COLLABORATOR_ENTERS_TO_LOCATION $C=Beto Byerley $L=oficina` y se notifica a VIGÍA.

VIGÍA

3. Recibe el evento contextual y lo procesa.
   - 3.a Encuentra las situaciones asociadas al evento contextual, entre ellas la situación "Colaborador cambia su ubicación".
   - 3.b Como la situación "Colaborador cambia su ubicación" es transitiva y no tiene condiciones, se notifica inmediatamente bajo el tópico `topico_general` (*Publish/Subscribe*).

Editor colaborativo de mapas mentales

4. El editor colaborativo de mapas mentales, previamente suscrito al `topico_general`, recibe el mensaje con el evento contextual `COLLABORATOR_ENTERS_TO_LOCATION`.
5. Realiza proceso de adaptación (cf. figura 4.18).
   - 5.a Se refleja la nueva ubicación del usuario en el área de conciencia grupal.
   - 5.b Se despliega un mensaje de notificación en la interfaz gráfica de usuario.

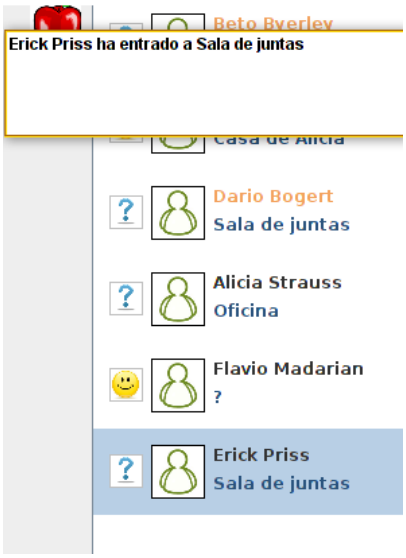


Figura 4.18: Adaptación del editor colaborativo ante la situación de "Colaborador cambia su ubicación", se refleja el cambio en la barra de colaboradores y se despliega un mensaje de notificación.

Situación: "Periodo de introducción"

El diagrama de secuencia mostrado en la figura 4.22 ilustra el proceso de adaptación de la situación "Periodo de introducción". Los pasos involucrados en este proceso para cada módulo son los siguientes:

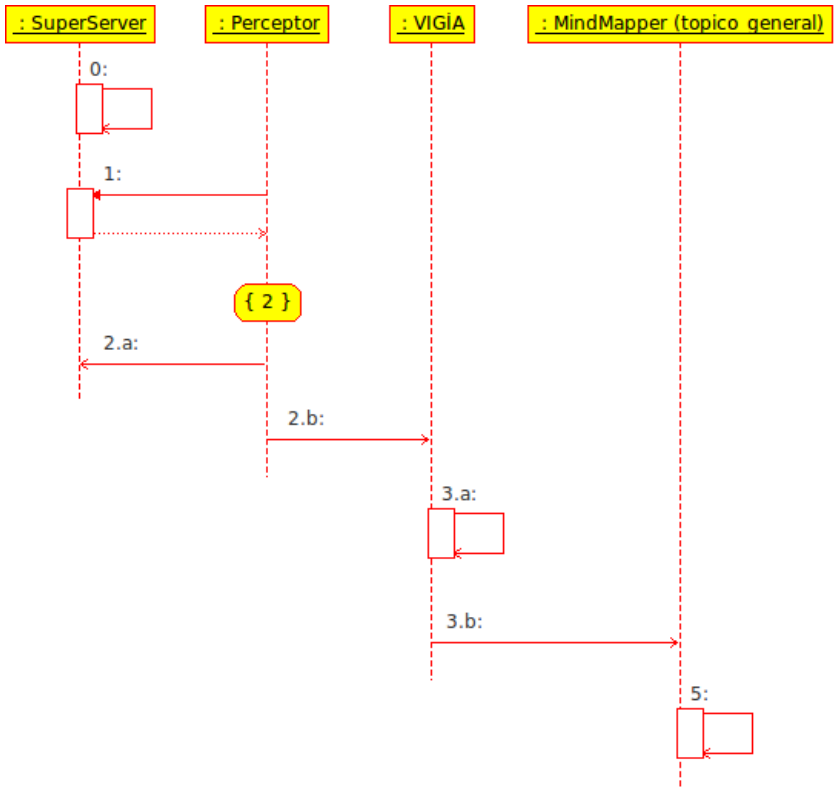
SuperServer

0. Iniciar SuperServer, pues es el módulo al que se le hacen peticiones sobre la base de datos.

Perceptor

1. Iniciar Perceptor, para lo cual se hacen peticiones a SuperServer para obtener la información necesaria de la base de datos.
2. En la pestaña Training Period de la GUI (cf. figura 4.20) se puede simular que un usuario deja de estar en periodo de introducción, e.g., Beto Byerley entra a periodo regular.
   - 2.a Se notifica a SuperServer para que cambie el estado del usuario.
   - 2.b Se produce el evento contextual, e.g., `COLLABORATOR_CHANGE_INTRODUCTORY_PERIOD_STATE $C=Beto Byerley` y se notifica a VIGÍA.

VIGÍA



0. Inicia SuperServer
1. Inicia Perceptor
2. Se simula evento contextual COLLABORATOR_ENTERS_TO_LOCATION
2.a. Se notifica a SuperServer para que cambie la ubicación del colaborador en la BD
2.b. Se notifica a VIGIA el evento contextual COLLABORATOR_ENTERS_TO_LOCATION $C=Beto Byerley $L=oficina
3.a. Encuentra las situaciones asociadas al evento contextual (LOCATION CHANGE)
3.b. LOCATION CHANGE es transitiva y sin condiciones, por lo que se publica inmediatamente bajo topico_general
5. Se adapta el sistema colaborativo

Figura 4.19: Diagrama de secuencia para la activación/adaptación de la situación "Colaborador cambia su ubicación"

3. Recibe el evento contextual y lo procesa.

   3.a Encuentra las situaciones asociadas al evento contextual, entre ellas la situación "Periodo de introducción".

   3.b Como la situación "Periodo de introducción" es permanente, se revisan las condiciones de activación para saber su estado (activo/inactivo), posteriormente se notifican ambos, la situación y sus estado, bajo los tópicos `topico_collaborator` y `topico_general` (*Publish/Subscribe*).

Editor colaborativo de mapas mentales

4. El editor colaborativo de mapas mentales, previamente suscrito al `topico_general` y/o al `topico_collaborator`, recibe el mensaje relativo a la situación "Periodo de introducción" y su respectivo estado (activo/inactivo).

5. Realiza proceso de adaptación (cf. figura 4.21)

   5.a Se despliega un mensaje de notificación en la interfaz gráfica de usuario.

   5.b Cambiar el ícono del colaborador de acuerdo al estado de la situación.

   5.c Se activa o desactiva la funcionalidad consistente en mostrar periódicamente información relevante para el usuario, cuando está en periodo de introducción.

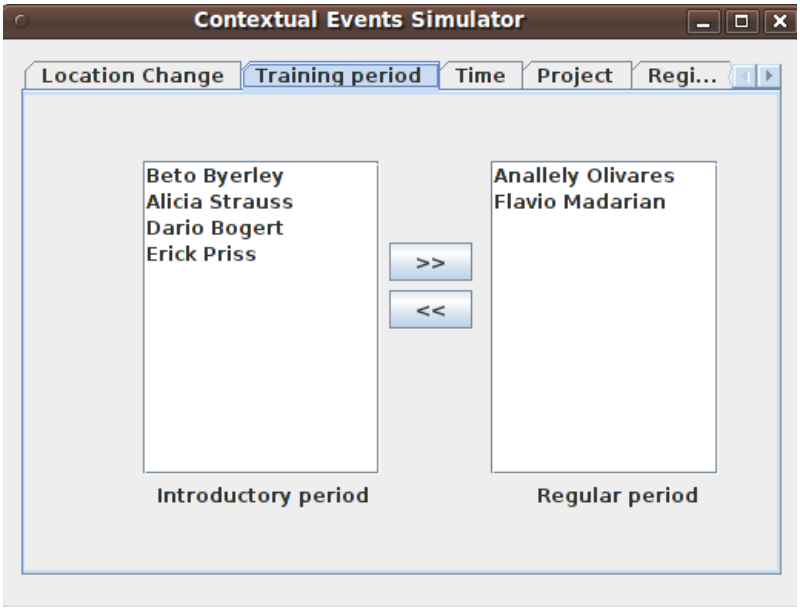


Figura 4.20: Pestaña Training Period en la interfaz gráfica de usuario del Perceptor

(a) Despliegue de un mensaje de notificación

(b) Se ajusta el ícono izquierdo de acuerdo al estado de la situación

(c) Si la situación está activa se muestra cada determinado tiempo información relevante para el usuario

Figura 4.21: Adaptación del editor colaborativo ante la situación "Periodo de introducción"

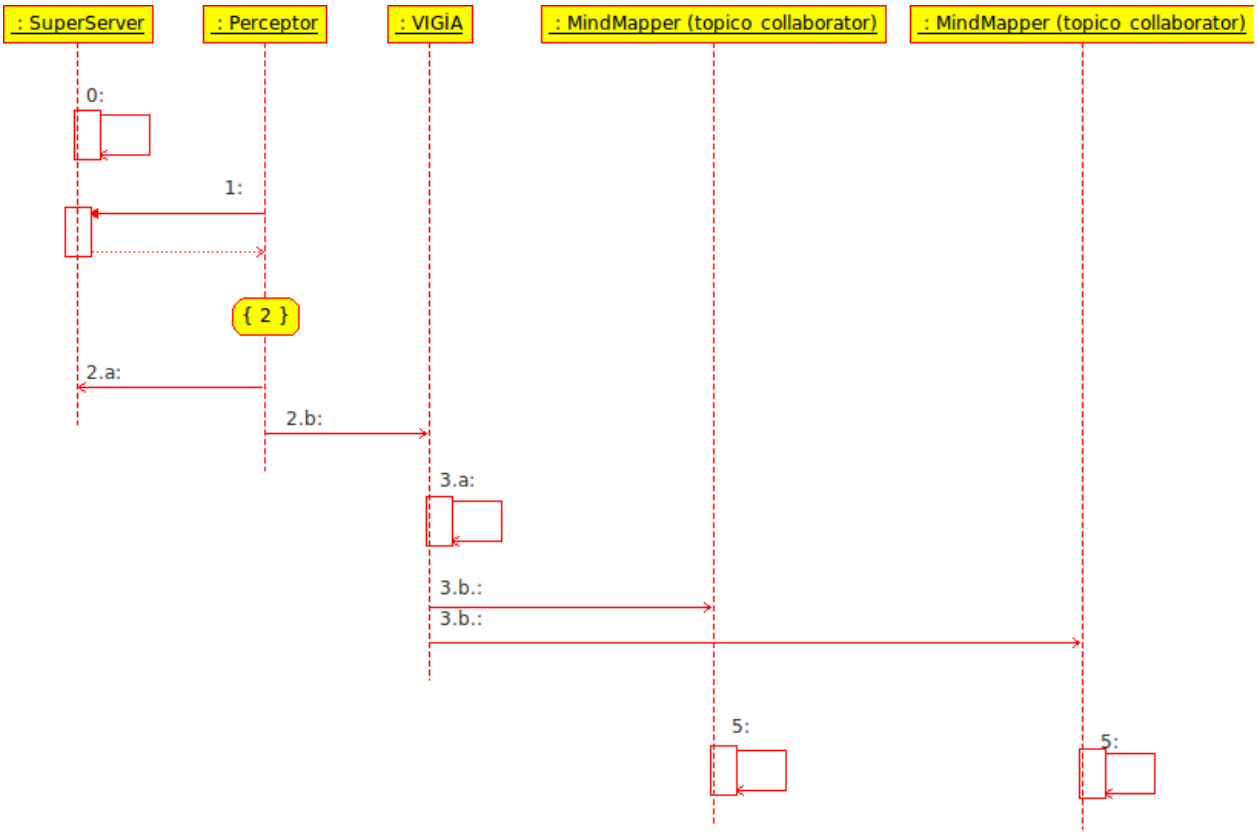


Figura 4.22: Diagrama de secuencia para la activación/adaptación de la situación "Periodo de introducción"

Situación: "Hora de comer"

El proceso de adaptación es análogo al de la situación "Periodo de introducción" (cf. subsección 4.2.2) pero con los eventos contextuales, parámetros, tópicos y condiciones correspondientes a la situación (cf. Apéndice B). Las reglas de adaptación implementadas para esta situación son las siguientes (cf. figura 4.23):

1. Se despliega un mensaje de notificación en la interfaz gráfica de usuario.

2. Se muestra u oculta un ícono que representa la situación en la barra de situaciones.

(a) Despliegue de un mensaje de notificación

(b) Ícono correspondiente cambia en la barra de situaciones

Figura 4.23: Adaptación del editor colaborativo ante la situación "Hora de comer"

Las situaciones "Intervalo de hora de entrada" e "Intervalo de hora de salida" muestran la misma adaptación en el sistema colaborativo.

Situación: "Proyecto en ruta crítica"

El proceso de adaptación es análogo al de la situación "Periodo de introducción" (cf. subsección 4.2.2) pero con los eventos contextuales, parámetros, tópicos y condiciones correspondientes a la situación (cf. Apéndice B). Las reglas de adaptación implementadas para esta situación son las siguientes (cf. figura 4.24):

1. Se despliega un mensaje de notificación en la interfaz gráfica de usuario.

2. Se reordena la barra de colaboradores: en caso de que la situación se active se colocan hasta arriba y en anaranjado los colaboradores con los que se necesita más interacción.

(a) Los colaboradores no están ordenados de manera específica

(b) Al activarse la situación se muestra un mensaje de notificación

(c) Y se reordenan los elementos de la barra de colaboradores

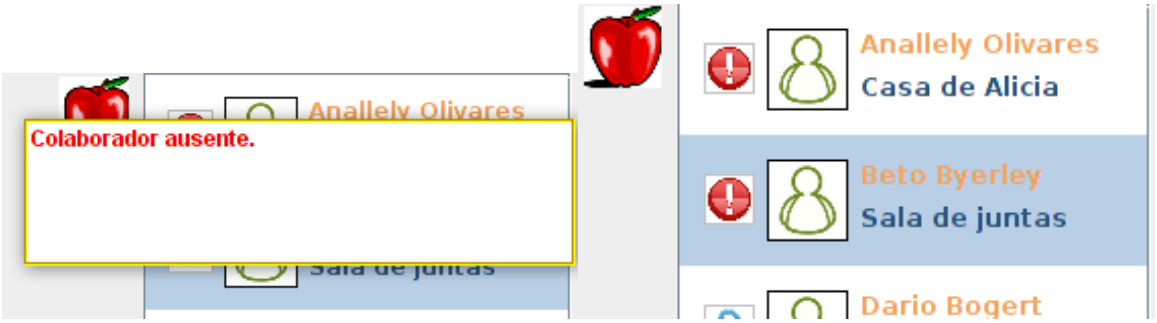
Figura 4.24: Adaptación del editor colaborativo ante la situación “Proyecto en ruta crítica”

Situación: "Colaborador ausente"

El proceso de adaptación es análogo al de la situación "Periodo de introducción" (cf. subsección 4.2.2) pero con los eventos contextuales, parámetros, tópicos y condiciones correspondientes a la situación (cf. Apéndice B). Las reglas de adaptación implementadas para esta situación son las siguientes (cf. figura 4.25):

1. Se despliega un mensaje de notificación en la interfaz gráfica de usuario.

2. Se ajusta el ícono izquierdo al colaborador de acuerdo al estado de la situación.



(a) Despliegue de un mensaje de notificación

(b) Se ajusta el ícono izquierdo de acuerdo al estado de la situación

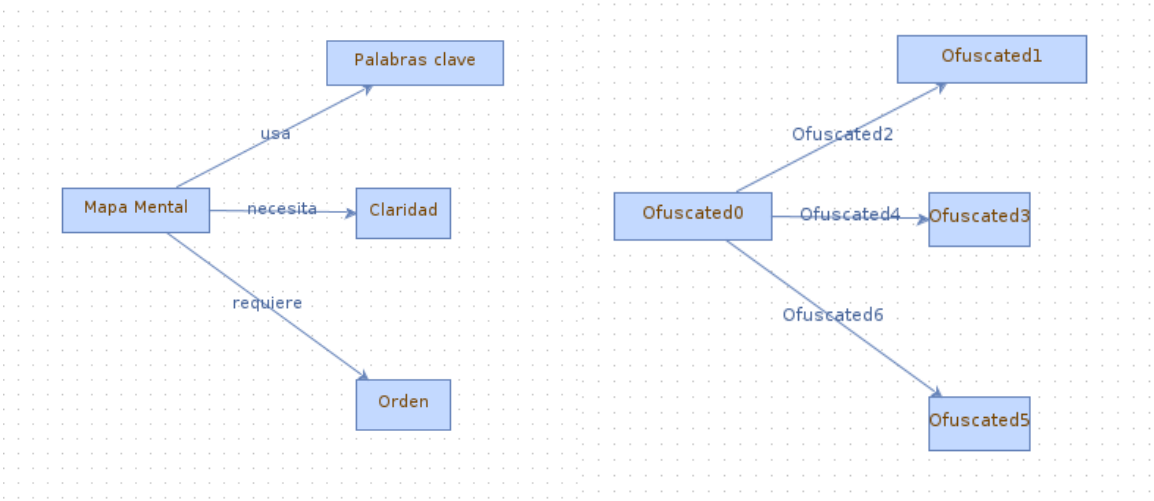
Figura 4.25: Adaptación del editor colaborativo ante la situación "Colaborador ausente"

Situación: "Persona externa detectada"

El proceso de adaptación es análogo al de la situación "Periodo de introducción" (cf. subsección 4.2.2) pero con los eventos contextuales, parámetros, tópicos y condiciones correspondientes a la situación (cf. Apéndice B). Las reglas de adaptación implementadas para esta situación son las siguientes (cf. figura 4.26):

1. Se ofusca el contenido del documento sobre el que se está trabajando si es que la situación se activa.

2. En el caso de que se desactive la situación, el documento vuelve a mostrarse de manera normal.



(a) Vista normal del documento

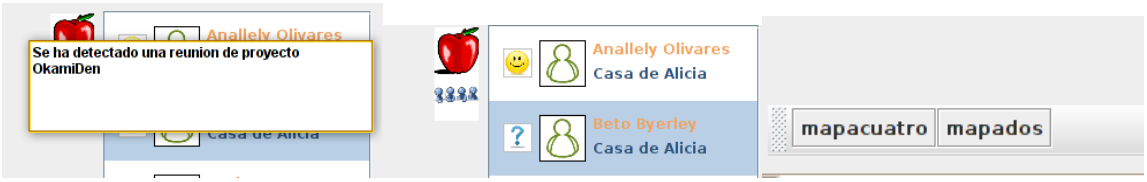
(b) Al detectarse una persona externa se ofusca el contenido del documento

Figura 4.26: Adaptación del editor colaborativo ante la situación "Persona externa detectada"

Situación: "Reunión"

El proceso de adaptación es análogo al de la situación "Periodo de introducción" (cf. subsección 4.2.2) pero con los eventos contextuales, parámetros, tópicos y condiciones correspondientes a la situación (cf. Apéndice B). Las reglas de adaptación implementadas para esta situación son las siguientes (cf. figura 4.27):

1. Se despliega un mensaje de notificación en la interfaz gráfica de usuario.
2. Se muestra u oculta un ícono que representa la situación en la barra de situaciones.
3. Se facilita el acceso a los documentos relevantes para la reunión.



(a) Despliegue de un mensaje de notificación

(b) Se muestra u oculta un ícono representando la situación en la barra de situaciones

(c) Se facilita el acceso a los documentos relevantes para la reunión

Figura 4.27: Adaptación del editor colaborativo ante la situación "Reunión"

Capítulo 5

Conclusiones y trabajo futuro

En este capítulo se hace una recapitulación de la problemática atacada por la arquitectura contextual propuesta (cf. sección 5.1). En seguida, se presentan las conclusiones derivadas del trabajo de investigación expuesto (cf. sección 5.2). Finalmente se discuten, como trabajo futuro, algunas limitaciones y extensiones del trabajo realizado (cf. sección 5.3).

5.1 Resumen de la problemática

La tesis descrita en este documento se enfoca en dar solución a los siguientes dos problemas, detectados al realizar el estudio de la literatura científica relacionada con la investigación (cf. capítulo 2):

1. La investigación actual en sistemas contextuales considera principalmente elementos físicos, como la ubicación y la plataforma. Muchas otras variables no se tienen en cuenta, en especial variables lógicas, las cuales son fundamentales al considerar el contexto en los entornos colaborativos [Brézillon et al., 2008].

2. Las arquitecturas y *frameworks* para desarrollar sistemas contextuales se han centrado principalmente en el contexto de un solo usuario, por lo que el contexto de un grupo de colaboradores sigue siendo un tema inexplorado. De ahí que algunas cuestiones importantes permanecen abiertas:

   - Definición de contexto de uso para entornos colaborativos.
   - Propuesta de una forma de combinar los estados físicos y lógicos de los miembros del grupo.
   - Especificación del tipo de situaciones que pueden ser explotadas en entornos colaborativos.

A través del trabajo de investigación realizado, se ha logrado presentar una solución para estos problemas, principalmente a través de: 1) la propuesta de una arquitectura contextual que considera estos aspectos, 2) la aportación de una definición de contexto de uso para entornos colaborativos (así como varios de sus elementos, e.g., evento contextual, situación, etc.) y 3) proposiciones de cómo explotar algunas situaciones en los sistemas colaborativos. Más detalles se presentan en la siguiente sección.

5.2 Conclusiones

En este documento de tesis se presentó una arquitectura de cuatro fases para soportar el contexto de uso en sistemas colaborativos:

1. Percepción de eventos contextuales.
2. Detección de situaciones.
3. Comunicación de situaciones.
4. Adaptación del sistema.

Dicha arquitectura contextual es capaz de soportar cambios tanto en variables físicas como lógicas en entornos colaborativos.

Las principales contribuciones del presente trabajo de investigación son las siguientes:

- Se proporcionó una definición de **contexto de uso en entornos colaborativos**, basada en la definición de contexto proporcionada por Coutaz y Rey [Coutaz and Rey, 2002], quedando nuestra definición de la siguiente manera:

  > "Cuando nos referimos al contexto en entornos colaborativos, en realidad hablamos del contexto de cada entidad presente, donde el contexto es considerado como la composición de múltiples situaciones en un cierto periodo de tiempo".

- Partiendo de la premisa de que las personas y los dispositivos que se encuentran en la misma ubicación están sujetos a las mismas variables físicas, se propuso el elemento **perceptor**, cuya función es la de monitorear las variables físicas presentes en la ubicación y determinar cuándo y qué eventos contextuales deben producirse.

- Se determinó que las situaciones son producto de eventos contextuales físicos y lógicos, por lo que una situación tiene que ser evaluada solo en el momento en que un evento contextual ha sido producido.

- Se propuso un modelo para representar situaciones con base en sus propiedades: nombre, entidad relacionada, observadores, eventos contextuales y condiciones. Al mismo tiempo las situaciones se clasifican en dos grupos: transitivas y permanentes; su definición y características fueron proveídas en este documento.
- Un algoritmo para la detección de cambios en el estado de las situaciones fue desarrollado.
- Partiendo de la premisa de que los usuarios no están interesados en todos los cambios de estado de las situaciones, sino sólo en aquellos que los afectan directamente, se propuso el uso del esquema de mensajes *Publish/Subscribe* para la comunicación de cambios en las situaciones.
- Se concluyó que el contexto está compuesto de todas las situaciones presentes, i.e., es posible que múltiples situaciones co-existan al mismo tiempo.

Algunas ideas y escenarios sobre cómo explotar el contexto en sistemas colaborativos también fueron propuestos. De hecho, el diseño de la arquitectura se basa en el análisis de estos escenarios.

5.3 Trabajo futuro

En el estado actual de la presente propuesta, las reglas de adaptación del sistema corresponden a reglas de la forma **if** *situación detectada* **then** *adaptación subsecuente*, i.e., un cambio específico en el sistema es realizado de acuerdo a la situación detectada. La **adaptabilidad dinámica** tiene que ser explorada para aprovechar al máximo las características de adaptación del sistema, en lugar de delimitarla por medio de reglas estáticas. Algoritmos de aprendizaje automático podrían utilizarse para este propósito.

El objetivo de adaptabilidad no solo es aumentar la efectividad del sistema, sino también la efectividad del usuario y del grupo al interactuar con el sistema, por lo que las reglas dinámicas deben considerar la **retroalimentación del usuario y sus colegas** como un factor importante.

Algunos otras mejoras y/o puntos por investigarse se presentan a continuación:

- Continuar analizando y proponiendo situaciones contextuales que pueden ser aprovechadas para aumentar la efectividad de los sistemas colaborativos.
- Al analizar un entorno colaborativo específico se podrían obtener ideas sobre la mejor forma de implementar la arquitectura contextual propuesta y/o aportar nuevos conceptos y mejoras al diseño. Por ejemplo, se podrían analizar las necesidades particulares de un grupo de trabajo en cierta organización del mundo real.

- Actualmente, las situaciones se modelan directamente en líneas de código, i.e., una modificación en cierta situación implica la modificación del código fuente. Sería conveniente proponer una forma de representación intermedia, e.g., un documento XML en donde se describan la situaciones a través de etiquetas, elementos, atributos, etc., de tal manera que el código fuente se encargue de interpretar el documento XML y que la modificación a una situación implique modificar el documento XML, mas no las líneas de código fuente (cf. figura 5.1).

- La activación/desactivación de una situación pueden ocasionar a su vez la activación/desactivación de alguna otra, i.e., son fuentes potenciales de eventos contextuales, que no fueron consideradas en el presente trabajo de investigación.

- Actualmente la implementación de la fase de percepción de eventos contextuales consiste en un simulador, por lo que queda como trabajo futuro hacer uso de sensores físicos y sistemas de información reales para una implementación más completa de esta fase.

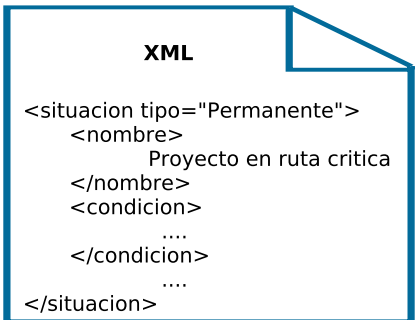


Figura 5.1: Modelado de una situación a través de un documento XML

Apéndice A

Monitoreo de la ubicación

Para conocer la ubicación de personas y dispositivos no basta un sensor, sino que se necesita de una infraestructura más compleja. La tabla A.1 muestra algunos sistemas relevantes para monitorear la ubicación.

Sistema	Funcionamiento
Active Badge [Want et al., 1992]	*Badges* y sensores IR
BAT [Harter et al., 1999]	Transmisores y receptores ultrasónicos
RADAR [Bahl and Padmanabhan, 2000]	Sistema basado en RF y triangulación
Smart Floor [Orr and Abowd, 2000]	Análisis de la fuerza de pisada

Tabla A.1: Sistemas para detectar la ubicación de personas y dispositivos

A.1 Active Badge

Se trata de un sistema para localizar personas en ambientes oficinales. Se basa en una tarjeta de identificación (*badge*) que debe ser portada por los usuarios (cf. figura A.1) y que se encarga de proveer información sobre su ubicación a un servidor central.

La tarjeta emite una señal infrarroja cada 15 segundos, la cual es capturada por una red de sensores debidamente localizada en el edificio. Un servidor maestro, también conectado a la red de sensores, monitorea las señales y procesa los datos para determinar las ubicaciones de las tarjetas y hacerlas disponibles a los clientes. El sistema provee ubicaciones simbólicas que representan la habitación en donde la tarjeta se encuentra, e.g., la sala de juntas.

Una de las desventajas de una señal transmitida con tan poca frecuencia es que la ubicación de la tarjeta solo se conoce, en el mejor de los casos, con una ventana de tiempo de 15 segundos de diferencia. Sin embargo, debido a que en general una persona

tiende a moverse relativamente poco en un edificio de oficinas, la información obtenida a través del sistema proporciona una gran precisión [Hightower and Borriello, 2001].

Figura A.1: Tarjeta usada en el sistema Active Badge para la ubicación de personas

A.2 Sistema BAT

En este sistema los usuarios y dispositivos también usan un identificador (cf. figura A.2) que envía, en respuesta a una petición, un pulso ultrasónico a una red de receptores montada en el techo de las habitaciones. La red de receptores calcula la distancia hacia la tarjeta identificadora al medir el tiempo de respuesta entre la petición y la recepción del pulso ultrasónico. La distancia computada es enviada entonces a un servidor central que se encarga de determinar la ubicación.

Figura A.2: Identificador usado en el sistema BAT para la ubicación de personas y dispositivos

El sistema reporta tener una precisión de hasta 9 cm de la posición verdadera para el 95 % de las mediciones. Cabe mencionar que necesita de una red de sensores en cada habitación, además de ser muy sensible al correcto posicionamiento de los sensores, por lo que la escalabilidad, facilidad de desarrollo y el costo son las desventajas de este enfoque [Hightower and Borriello, 2001].

A.3 Sistema RADAR

Se trata de un sistema que analiza desde un punto de acceso (PA) la intensidad y la relación señal-ruido de las señales emitidas por dispositivos inalámbricos y a partir de esta información calcula la posición 2D dentro de un edificio.

Los autores notaron que la fuerza de la señal varía mientras el usuario camina dentro del edificio portando un dispositivo móvil: se vuelve más intensa mientras más cerca se encuentre de un punto de acceso y se debilita mientras se aleja.

El enfoque de RADAR ofrece la ventaja de que requiere pocas estaciones centrales y usa la misma infraestructura inalámbrica de propósito general existente en el edificio. Sin embargo, sufre de dos desventajas: 1) el dispositivo a ser monitoreado debe tener soporte para redes LAN inalámbricas y 2) no es un problema trivial generalizar su funcionamiento para edificios con múltiples pisos.

Se tienen dos versiones del sistema: análisis de escenas y triangulación. La primera de ellas es capaz de lograr una precisión de hasta 3 metros o menos de la posición verdadera en el 50 % de las mediciones, mientras que la segunda tiene una precisión de 4.3 metros con la misma probabilidad. A pesar de que la técnica de análisis de escenas tiene mejor precisión, pequeños cambios en el ambiente pueden necesitar la reconstrucción de un mapa predefinido auxiliar en el análisis de las señales [Hightower and Borriello, 2001].

A.4 Smart Floor

Se trata de un mecanismo que no está basado en ninguna tarjeta o dispositivo de identificación, sino en el análisis de la fuerza de pisada de las personas (cf. figura A.3). Para realizar este análisis se necesita de una infraestructura que consiste en sensores de fuerza ubicados en el piso de lugares estratégicos, e.g., la entrada de las habitaciones.

La precisión de este sistema es de un 93 %, sin embargo solo funciona para grupos pequeños de trabajo (hasta 15 personas). La principal ventaja de esta aproximación consiste en que es transparente y natural para los usuarios, pero solo es útil para monitorear la ubicación de las personas, no dispositivos.

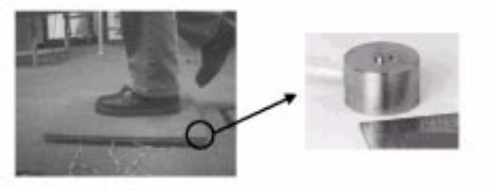


Figura A.3: El sistema Smart Floor funciona con base en el perfil de pisada de los usuarios

Apéndice B

Modelado de algunas situaciones

Colaborador cambia su ubicación	
Tipo	Transitiva.
Observadores	Todos.
Eventos contextuales	- Colaborador entra a una ubicación. - Colaborador sale de una ubicación.

Tabla B.1: Propiedades de la situación "Cambio de localización"

Periodo de introducción	
Tipo	Permanente.
Entidad relacionada	Usuario de reciente ingreso.
Observadores	Todos (en especial el usuario en periodo de introducción).
Eventos contextuales	- Nuevo colaborador registrado. - Periodo de introducción ha terminado.
Condiciones	*Activar:* Evento contextual "Nuevo colaborador registrado". *Desactivar:* Evento contextual "Periodo de introducción ha terminado".

Tabla B.2: Propiedades de la situación "Periodo de introducción"

Hora de salida	
Tipo	Transitiva.
Observadores	Todos.
Eventos contextuales	Hora de salida.

Tabla B.3: Propiedades de la situación "Hora de salida"

Proyecto en ruta crítica	
Tipo	Permanente.
Entidad relacionada	Proyecto que se encuentra en ruta crítica.
Observadores	Proyecto.
Eventos contextuales	- Alerta: proyecto en ruta crítica. - Estado del proyecto cambia a "entregado". - Estado del proyecto cambia a "cancelado". - Estado del proyecto cambia a "en desarrollo". - Fecha de entrega del proyecto actualizada.
Condiciones	*Activar:* Menos de n días para entregar el proyecto. *Desactivar:* Evento contextual "Estado del proyecto cambia a entregado". Evento contextual "Estado del proyecto cambia a cancelado". Más de n días para entregar el proyecto.

Tabla B.4: Propiedades de la situación "Proyecto en ruta crítica"

Colaborador ausente	
Tipo	Permanente.
Entidad relacionada	Colaborador ausente.
Observadores	Colegas del colaborador ausente.
Eventos contextuales	- Colaborador entra a una ubicación. - Colaborador ingresa al sistema. - Alerta: colaborador ausente. - Cambio en la situación "Proyecto en ruta crítica".
Condiciones	*Activar:* La presencia del colaborador ausente es requerida. *Desactivar:* El colaborador ausente ingresa al sistema. El colaborador ausente se detecta en cierta ubicación.

Tabla B.5: Propiedades de la situación "Colaborador ausente"

Persona externa detectada	
Tipo	Permanente.
Entidad relacionada	Ubicación donde se detecta a la persona externa.
Observadores	Ubicación donde se detecta a la persona externa.
Eventos contextuales	- Persona externa entra a la ubicación. - Persona externa sale de la ubicación.
Condiciones	*Activar:* Evento contextual "Persona externa entra a la ubicación". Número de personas externas es mayor o igual a una. *Desactivar:* Número de personas externas es cero.

Tabla B.6: Propiedades de la situación "Persona externa detectada"

Reunión	
Tipo	Permanente.
Entidad relacionada	Ubicación donde se lleva a cabo la reunión.
Observadores	Personas co-localizadas en la ubicación.
Eventos contextuales	- Colaborador entra a la ubicación. - Colaborador sale de la ubicación. - Persona externa entra a la ubicación. - Persona externa sale de la ubicación. - Un proyector se enciende. - Un proyector se apaga.
Condiciones	*Activar:* El número de personas co-localizadas debe ser mayor a uno y un proyector está encendido y las personas co-localizadas trabajan en el mismo proyecto.

Tabla B.7: Propiedades de la situación "Reunión"

Tazas de café se acaban	
Tipo	Transitiva.
Observadores	Aplicación responsable de enviar e-mails.
Eventos contextuales	- Usuario toma una taza. - Usuario regresa una taza. - El valor del umbral ha cambiado.
Condiciones	Número de tazas menor a un umbral establecido.

Tabla B.8: Propiedades de la situación "Tazas de café se acaban"

Ambiente seguro	
Tipo	Permanente.
Entidad relacionada	Ubicación.
Observadores	Personas co-localizadas en la ubicación.
Eventos contextuales	- Colaborador entra a la ubicación. - Colaborador sale de la ubicación. - Persona externa entra a la ubicación. - Persona externa sale de la ubicación. - Se ha programado una nueva reunión. - Se ha cambiado/cancelado una reunión. - Inician las horas de oficina. - Terminan las horas de oficina. - Cambio en el estado de la red detectado. - Se abre la puerta de la habitación. - Se cierra la puerta de la habitación.
Condiciones	Todo el personal está autorizado y una reunión está programada y son horas de oficina y se encuentra presente una autoridad y la seguridad de la red es apropiada y la puerta de la habitación está cerrada.

Tabla B.9: Propiedades de la situación “Ambiente seguro”

Publicaciones del autor

[Olivares et al., 2011] Olivares,A., Mendoza, S., and de Luca, A. (2011) An Architecture to Support Context of Use in Groupware Systems. 8th International Conference on Electrical Engineering, Computing Science and Automatic Control *(CCE)*, pages 807–812, Mérida, Yucatán. México October 26-28, 2011. IEEE.

Bibliografía

[Abowd et al., 1999] Abowd, G. D., Dey, A. K., Brown, P. J., Davies, N., Smith, M., and Steggles, P. (1999). Towards a better understanding of context and context-awareness. In Gellersen, H.-W., editor, *HUC*, volume 1707 of *Lecture Notes in Computer Science*, pages 304–307. Springer.

[Al-Rwais and Al-Muhtadi, 2009] Al-Rwais, S. and Al-Muhtadi, J. (2009). A context-aware access control model for pervasive environments. In Xiang, Y., Lopez, J., Wang, H., and Zhou, W., editors, *NSS*, pages 425–430. IEEE Computer Society.

[Bahl and Padmanabhan, 2000] Bahl, P. and Padmanabhan, V. N. (2000). Radar: An in-building rf-based user location and tracking system. In *INFOCOM*, pages 775–784.

[Baldauf et al., 2007] Baldauf, M., Dustdar, S., and Rosenberg, F. (2007). A survey on context-aware systems. *IJAHUC*, 2(4):263–277.

[Benson and Alder, 2011] Benson, D. and Alder, G. (2011). Jgraphx (jgraph 6) user manual. http://www.jgraph.com/doc/mxgraph/index_javavis.html.

[Brézillon et al., 2008] Brézillon, P., Borges, M. R. S., Pino, J. A., and Pomerol, J.-C. (2008). Context and awareness in group work lessons. Learned from three case studies. *Journal of Decision Systems*, 17(1):27–40.

[Calvary et al., 2004] Calvary, G., Coutaz, J., Dâassi, O., Balme, L., and Demeure, A. (2004). Towards a new generation of widgets for supporting software plasticity: The "Comet". In Bastide, R., Palanque, P. A., and Roth, J., editors, *EHCI/DS-VIS'04 Engineering Human Computer Interaction and Interactive Systems, Joint Working Conferences*, volume 3425 of *Lecture Notes in Computer Science*, pages 306–324, Hamburg, Germany, July 11-13, 2004. Springer.

[Calvary et al., 2001] Calvary, G., Coutaz, J., and Thevenin, D. (2001). A unifying reference framework for the development of plastic user interfaces. In Little, M. R. and Nigay, L., editors, *EHCI 2001 Engineering for Human-Computer Interaction, 8th IFIP International Conference*, volume 2254 of *Lecture Notes in Computer Science*, pages 173–192, Toronto, Canada, May 11-13, 2001. Springer.

[Cardoso and José, 2009] Cardoso, J. C. S. and José, R. (2009). A framework for context-aware adaptation in public displays. In Meersman, R., Herrero, P., and Dillon, T. S., editors, *OTM Workshops 2009 On the Move to Meaningful Internet Systems*, volume 5872 of *Lecture Notes in Computer Science*, pages 118–127, Vilamoura, Portugal, November 1-6, 2009. Springer.

[Chen, 2004] Chen, H. (2004). *An Intelligent Broker Architecture for Pervasive Context-Aware Systems*. PhD thesis, University of Maryland, Baltimore County.

[Coutaz and Calvary, 2008] Coutaz, J. and Calvary, G. (2008). HCI and software engineering: Designing for user interface plasticity. In *The Human Computer Interaction Handbook*, chapter 56, pages 1107–1125.

[Coutaz and Rey, 2002] Coutaz, J. and Rey, G. (2002). Foundations for a theory of contextors. In Kolski, C. and Vanderdonckt, J., editors, *CADUI Computer-Aided Design of User Interfaces III, Proceedings of the Fourth International Conference on Computer-Aided Design of User Interfaces*, pages 13–34, Valenciennes, France, May, 15-17, 2002. Kluwer.

[Crowley et al., 2002] Crowley, J. L., Coutaz, J., Rey, G., and Reignier, P. (2002). Perceptual components for context aware computing. In Borriello, G. and Holmquist, L. E., editors, *Ubicomp 2002 Ubiquitous Computing, 4th International Conference*, volume 2498 of *Lecture Notes in Computer Science*, pages 117–134, Göteborg, Sweden, September 29 - October 1, 2002. Springer.

[Dey et al., 1999] Dey, A. K., Salber, D., Abowd, G. D., and Futakawa, M. (1999). The conference assistant: Combining context-awareness with wearable computing. In *ISWC*, pages 21–28.

[Eugster et al., 2003] Eugster, P. T., Felber, P., Guerraoui, R., and Kermarrec, A.-M. (2003). The many faces of publish/subscribe. *ACM Comput. Surv.*, 35(2):114–131.

[Foundation, 2011] Foundation, A. S. (2011). Apache mina. http://mina.apache.org/.

[Gamma et al., 1995] Gamma, E., Helm, R., Johnson, R. E., and Vlissides, J. (1995). *Design Patterns: Elements of Reusable Object-Oriented Software*. Addison-Wesley, Reading, MA.

[Ghiani et al., 2009] Ghiani, G., Paternò, F., Santoro, C., and Spano, L. D. (2009). Ubicicero: A location-aware, multi-device museum guide. *Interacting with Computers*, 21(4):288–303.

[Grolaux et al., 2002] Grolaux, D., Roy, P. V., and Vanderdonckt, J. (2002). Flexclock, a plastic clock written in oz with the qtk toolkit. In Pribeanu, C. and Vanderdonckt, J., editors, *TAMODIA 2002 Task Models and Diagrams for User*

Interface Design: Proceedings of the First International Workshop on Task Models and Diagrams for User Interface Design, pages 135–142, Bucharest, Romania, 18-19 July 2002. INFOREC Publishing House Bucharest.

[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. *Int. J. Cooperative Inf. Syst.*, 19(1-2):71–120.

[Harter et al., 1999] Harter, A., Hopper, A., Steggles, P., Ward, A., and Webster, P. (1999). The anatomy of a context-aware application. In *Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*, MobiCom '99, pages 59–68, New York, NY, USA. ACM.

[Hightower and Borriello, 2001] Hightower, J. and Borriello, G. (2001). Location systems for ubiquitous computing. *IEEE Computer*, 34(8):57–66.

[Hussein et al., 2010] Hussein, T., Linder, T., Gaulke, W., and Ziegler, J. (2010). A framework and an architecture for context-aware group recommendations. In Kolfschoten, G. L., Herrmann, T., and Lukosch, S., editors, *CRIWG 2010 Collaboration and Technology - 16th International Conference*, volume 6257 of *Lecture Notes in Computer Science*, pages 121–128, Maastricht, The Netherlands, September 20-23, 2010. Springer.

[McKeever et al., 2009] McKeever, S., Ye, J., Coyle, L., and Dobson, S. (2009). A context quality model to support transparent reasoning with uncertain context. In Rothermel, K., Fritsch, D., Blochinger, W., and Dürr, F., editors, *QuaCon*, volume 5786 of *Lecture Notes in Computer Science*, pages 65–75. Springer.

[NokiaBetaLabs, 2010] NokiaBetaLabs (2010). Nokia situations. http://betalabs.nokia.com/apps/nokia-situations.

[Oracle, 2011a] Oracle (2011a). Class objectoutputstream. http://download.oracle.com/javase/1.5.0/docs/api/java/io/ObjectOutputStream.html.

[Oracle, 2011b] Oracle (2011b). Class objectoutputstream. http://download.oracle.com/javase/1.5.0/docs/api/java/io/ObjectInputStream.html.

[Oracle, 2011c] Oracle (2011c). What is swing? http://docs.oracle.com/javase/tutorial/ui/overview/intro.html.

[Orr and Abowd, 2000] Orr, R. J. and Abowd, G. D. (2000). The smart floor: a mechanism for natural user identification and tracking. In *CHI '00 extended abstracts on Human factors in computing systems*, CHI EA '00, pages 275–276, New York, NY, USA. ACM.

[Rama and Bishop, 2006] Rama, J. and Bishop, J. (2006). A survey and comparison of cscw groupware applications. In *Proceedings of the 2006 annual research*

conference of the South African institute of computer scientists and information technologists on IT research in developing countries, SAICSIT '06, pages 198–205, , Republic of South Africa. South African Institute for Computer Scientists and Information Technologists.

[Schilit et al., 1994] Schilit, B., Adams, N., and Want, R. (1994). Context-aware computing applications. *Mobile Computing Systems and Applications, IEEE Workshop on*, 0:85–90.

[Sendín and Collazos, 2006] Sendín, M. and Collazos, C. A. (2006). Implicit plasticity framework: A client-side generic framework for collaborative activities. In Dimitriadis, Y. A., Zigurs, I., and Gómez-Sánchez, E., editors, *CRIWG 2006 Groupware: Design, Implementation, and Use, 12th International Workshop*, volume 4154 of *Lecture Notes in Computer Science*, pages 219–227, Medina del Campo, Spain, September 17-21, 2006. Springer.

[Sendín et al., 2008] Sendín, M., López-Jaquero, V., and Collazos, C. A. (2008). Collaborative explicit plasticity framework: a conceptual scheme for the generation of plastic and group-aware user interfaces. *J. UCS*, 14(9):1447–1462.

[Strang and Linnhoff-Popien, 2004] Strang, T. and Linnhoff-Popien, C. (2004). A context modeling survey. In *In: Workshop on Advanced Context Modelling, Reasoning and Management, UbiComp 2004 - The Sixth International Conference on Ubiquitous Computing, Nottingham/England*.

[TIOBE-Software, 2011] TIOBE-Software (2011). Tiobe programming community index for november 2011. http://www.tiobe.com/.

[Vanderdonckt and González-Calleros, 2008] Vanderdonckt, J. and González-Calleros, J. M. (2008). Task-driven plasticity: One step forward with ubidraw. In Forbrig, P. and Paternò, F., editors, *TAMODIA/HCSE 2008 Engineering Interactive Systems, Second Conference on Human-Centered Software Engineering, HCSE 2008, and 7th International Workshop on Task Models and Diagrams*, volume 5247 of *Lecture Notes in Computer Science*, pages 181–196, Pisa, Italy, September 25-26, 2008. Springer.

[VMware, 2011] VMware (2011). Rabbitmq, messaging that just work. http://www.rabbitmq.com/.

[Want et al., 1992] Want, R., Hopper, A., Falcao, V., and Gibbons, J. (1992). The active badge location system. *ACM Trans. Inf. Syst.*, 10(1):91–102.