Universidad Nacional de La Plata

Facultad de Informática

Magister en Ingeniería de Software

Modelado de Sistemas Colaborativos

Autor: Luis Mariano Bibbó

Director: Claudia Pons

Co-Director: Gustavo H. Rossi

Dedicatoria:

A mis padres, hermanos

y a mis amores Luján, Paloma y Emilio

Agradecimientos:

A todos los amigos que conocí en el Lifia, desde el primero al último. Con cada uno de ellos aprendí muchas cosas técnicas y humanas.

ÍNDICE GENERAL

1. Índice de ilustraciones.....5
2. Tabla de abreviaciones.....7
3. Introducción.....9
3.1. Objetivos de la tesis.....12
3.2. Estructura de la tesis.....13
4. Definiciones y conceptos básicos.....15
4.1. Conceptos principales.....16
5. Comparación con las reuniones presenciales.....25
6. Conceptualización de sistemas groupware.....29
6.1. Comunicación, colaboración y coordinación.....29
6.2. Espectro de sistemas colaborativos 31
7. Justificación del modelo espacial.....37
8. Estado del arte.....48
8.1. Un poco de historia.....48
8.2. Ejemplos de aplicaciones.....50
9. Lenguaje de modelado de sistemas colaborativos (CSSL - Collaborative Software System Language).....78
9.1. ¿Por qué es necesario un metamodelo?.....78
9.2. Niveles de abstracción de la OMG.....79
9.3. Metamodelo.....82
10. Método de especificación.....113
10.1. Introducción:.....113
10.2. Descripción general del método.....116
10.3. Awareness.....129
11. Conclusiones.....133
11.1. Pasos a seguir.....135

12. Referencias.....138

1. Índice de ilustraciones

Ilustración 1: Sistema de Noticias.....52
Ilustración 2: Sistema de Chat.....54
Ilustración 3: Sistema de tormenta de ideas.....56
Ilustración 4: Compartir aplicaciones monousuarias.....58
Ilustración 5: Pizarra compartida.....60
Ilustración 6: Hipermedia colaborativo.....66
Ilustración 7: Colaboración en la Web.....70
Ilustración 8: Agendas compartidas.....72
Ilustración 9: Edición colaborativa.....74
Ilustración 10: Niveles de abstracción de la OMG.....81
Ilustración 11: Diagrama de paquetes del metamodelo de sistemas colaborativos.....82
Ilustración 12: Estructura del paquete Kernel.....83
Ilustración 13: Diseño de la jerarquía de los elementos colaborativos – Kernel.....84
Ilustración 14: Elementos Colaborativos del Workspace.....88
Ilustración 15: Asociaciones entre elementos colaborativos.....88
Ilustración 16: Diagrama de clases del paquete Protocols.....96
Ilustración 17: Casos de Uso.....100
Ilustración 18: Diagrama de la sala de reunión.....101
Ilustración 19: Modelo de la sala cafetería.....102
Ilustración 20: Modelo de la sala de capacitación.....102
Ilustración 21: Modelo de la sala de capacitación alternativo.....103
Ilustración 22: Diagrama de la sala de multimedia.....103
Ilustración 23: Modelo de roles.....104
Ilustración 24: Diagrama de proceso colaborativo.....106
Ilustración 25: Diagrama de protocolo de sesión.....107

Ilustración 26: Vital; sistema de aprendizaje colaborativo.....120
Ilustración 27: Vital; modelo de roles del sistema.....121
Ilustración 28: Especificación de Entorno Colaborativos (Espacios).....121
Ilustración 29: Notación general de protocolo entre sesiones.....122
Ilustración 30: Crocodile; modelado de proceso colaborativos.....124
Ilustración 31: Especificación de Procesos Colaborativos.....125
Ilustración 32: Notación general de protocolos estados de una sesión.....127
Ilustración 33: Ejemplo de Protocolo OneTwoChat.....129
Ilustración 34: Notación de Awareness en los diferentes niveles.....132
Ilustración 35: Layout del plugin para diseñar sistemas colaborativos.....136

2. Tabla de abreviaciones

Abreviación	Significado
CSSL	Collaborative Software System Language
CSCW	Computer Supported Cooperative Work
CSCL	Computer Supported Cooperative Learning
EMF	Eclipse Modelling Framework
DSL	Domain Specific Language
MOF	Meta Object Facility
GMF	Graphical Modeling Framework
TMF	Textual Modeling Framework
JET	Java Emitter Templates
JMerge	Java Merge
MDD	Model-Driven Development
UML	Unified Modeling Language
PSEE	Process-Centered *Software* Engineering Environments

3. Introducción

En este trabajo nos proponemos dar cuenta de los diferentes aspectos vinculados con sistemas que integran la participación en un mismo proyecto a muchos usuarios que pueden encontrarse en distintos lugares en diversas estaciones de trabajo conectados a través de una red. La tecnología involucrada se la denomina groupware, sistemas groupware o sistemas colaborativos [1], [2]. Asimismo presentaremos un lenguaje de dominio específico DSL que se usará para describir a los sistemas colaborativos - Collaborative Software System Language y que llamaremos CSSL [3] - con capacidad para enfrentar los problemas más importantes que esta tecnología aún presenta y para diseñar ambientes groupware teniendo en cuenta las demandas requeridas (obtener sistemas groupware integradores, flexibles y altamente productivos), y la complejidad tecnológica que implica desarrollar ambientes con estas características.

Típicamente los sistemas colaborativos fueron concebidos para compartir información y coordinar actividades relacionadas en un proyecto. Algunas de las características básicas de los sistemas groupware son compartir eventos de un calendario compartido, una lista de distribución para acceder a información compartida, un repositorio para compartir archivos o herramientas de comunicación como un Chat o Foro para comunicar a un grupo. Este tipo de aplicaciones son especialmente útiles en los casos en que el grupo de usuarios no se encuentran en el mismo lugar y necesitan compartir información e ideas para llegar a un objetivo común.

Para entender mejor los sistemas colaborativos, veamos aquellos que no lo son. Encontramos entonces sistemas donde el usuario interactúa solamente con el sistema -ya sea que el usuario acceda a una base de datos, prepara algún documento o interactúa con un video juego- incluso en los sistemas de información usado por muchos usuarios que no provean interacción entre ellos. De aquí vemos que el requerimiento principal de los

sistemas colaborativos es precisamente el soportar la interacción entre los usuarios del sistema como una actividad grupal y no como actividades en un contexto individual. Para satisfacer este requerimiento, los sistemas colaborativos, tendrán que cubrir tres áreas centrales la comunicación, coordinación y colaboración.

La comunicación basada en computadora ha mejorado mucho en los últimos años. Inicialmente existían las herramientas asincrónicas de envío de correo electrónico (solamente de texto) o los sistemas de noticias (sistemas de news) basados en mensajes de texto. Hoy hay una gran variedad de herramientas que permiten comunicar a un grupo de usuarios en forma sincrónica y enviando distintos tipos de información texto, sonido, imágenes, etc. Igualmente se esperan muchas mejoras en este aspecto, en especial de integración con otros dispositivos, en especial dispositivos móviles.

Por otro lado la efectiva colaboración requiere que los usuarios accedan a información compartida. Muchos sistemas (sistemas de bases de datos en general) tienden a aislar a los usuarios permitiendo acceder a los usuarios a una base de datos común, pero ninguno de ellos tiene la percepción de que hay otros usuarios en el mismo sistema. Por ejemplo no reciben notificación de cambios ni de que cosa están haciendo los otros usuarios en el sistema.

Finalmente la efectiva colaboración y comunicación puede ser mejorada con una buena coordinación. Sin coordinación los miembros del grupo podrán entrar en conflicto accediendo a los mismos recursos compartidos o tendrán que hacer un esfuerzo extra de comunicación para organizar sus actividades.

La etapa de implementación de Sistemas Colaborativos suele estar soportada por diversas herramientas y frameworks en general Orientados a Objetos [19], [20] pero no hay una manera unificada de especificar y diseñar las aplicaciones sobre estos frameworks. Esto dificulta la concepción de este tipo de sistemas debido, precisamente a la falta de modelo que permita representar los elementos principales de las aplicaciones colaborativas.

La construcción de Sistemas Colaborativos es una es una tarea muy compleja por el hecho de que involucra la interacción intensiva entre actores que están dispersos en diferentes lugares. Independientemente de la arquitectura sobre la cual esté construido, el sistema deberá controlar la participación de cada actor sobre el conjunto de objetos compartidos y mantener actualizado a todos los participantes de los cambios producidos. La adopción de procesos que guíen la construcción de este tipo de sistemas es fundamental porque sostiene las prácticas repetibles y técnicas que organizan el desarrollo y mejoran la calidad de los productos. Distintos procesos pueden diferir en definir cuando hay que realizar una actividad o que parte del sistema estas actividades desarrollan y el rigor y grado de formalismo en el que cada artefacto debe ser construido. En el contexto de la ingeniería de software el desarrollo guiado por modelos (Model Driven Development approach - MDD) [4], [5], [6] surge como un paradigma que cambia el desarrollo basado en código por el desarrollo basado en modelos. Este enfoque promueve la sistematización y automatización de la construcción de artefactos de software. Los modelos son considerados los primeros productos en el desarrollo de software, y el conocimiento de los desarrolladores está encapsulado en las transformaciones de esos modelos.

Para aplicar exitosamente el paradigma MDD al desarrollo de sistemas colaborativos en primer lugar tenemos que ser capaces de crear modelos abstractos de los sistemas colaborativos. Estos modelos, pueden ser escritos en UML [7]; sin embargo esta tarea suele ser para los desarrolladores ciertamente abrumadora por el hecho de cubrir la gran distancia que hay entre la semántica de UML y la del dominio de los sistemas colaborativos. Dicho de otra manera; los lenguajes de propósito general, como el UML, no expresan de una forma directa los conceptos del dominio ni su semántica. Es así que surge la necesidad de contar con un lenguaje preciso que permita expresar de forma concisa y sencilla los conceptos del dominio.

Existen varias teorías que permiten construir lenguajes (muchos de ellos como extensión de UML) que expresan algunas características de los sistemas colaborativos, como lenguajes para modelar hypermedia y sistemas distribuidos presentado en [9], la notación

para modelar datos compartidos en sistemas colaborativos descripto en [10], el lenguaje para especificar sistemas distribuidos introducido en [11], la extensión de UML llamado AUML [12] para describir protocolos de comunicación entre agentes y la extensión de aplicaciones interactivas definida en [13] entre otras. Sin embargo estos lenguajes ofrecen una notación específica para modelar un número particular de funcionalidad colaborativa, ninguno cubre el dominio de una forma exhaustiva, dejando de lado algunos conceptos del dominio. Por otro lado estos lenguajes no tienen la intensión de servir como primer paso para aplicar derivación de modelos en la modalidad MDD para desarrollar sistemas colaborativos.

3.1. Objetivos de la tesis

En este trabajo perseguiremos un objetivo principal que consiste en facilitar la concepción y el desarrollo de sistemas colaborativos. En este sentido es que identificamos algunos sub-objetivos que pueden ser considerados resultados parciales necesarios para lograr el objetivo final. Podemos mencionar que uno de los sub-objetivos es el de conceptualizar los sistemas groupware, entender para que pueden ser usados, identificar los conceptos principales que componen estos sistemas. A partir de esto se va a ver facilitada la comunicación entre los desarrolladores groupware.

Otro de los sub-objetivos es obtener un lenguaje de dominio específico DSL (Domain Specific Language) para describir sistemas colaborativos que llamaremos CSSL (Collaborative Software System Language) [3]. Este lenguaje fue diseñado como una extensión de UML usando el mecanismo de metamodelado, que es una técnica bastante común para especificar lenguajes de dominio específico. En definitiva será una descripción del sistema escrito en un lenguaje bien formado, el cual esta preparado para ser interpretado automáticamente. Además, éste lenguaje posee una sintaxis abstracta formalizada que permitirá entre otras cosas definir más de una sintaxis concreta a partir de ella, lo cual aumenta la legibilidad y flexibilidad del lenguaje.

Finalmente otro sub-objetivo que nos planteamos es el de obtener una guía que facilite la construcción de estos sistemas. Dentro del paradigma MDD entendemos que es imprescindible trabajar en las etapas iniciales del proceso de desarrollo que es donde se conciben estas aplicaciones. Los modelos subsiguientes entre ellos de diseño, codificación y en definitiva la implementación del sistema se podrán ir derivando a partir de los modelos previos. Por ello es que proponemos un método que permite especificar los sistemas colaborativos utilizando algunas sintaxis concretas obtenidas a partir del lenguaje CSSL.

3.2. Estructura de la tesis

Para lograr los objetivos planteados es que estructuramos la tesis de la siguiente manera: La primera parte del trabajo está dedicada a conceptualizar los sistemas colaborativos, definiendo los conceptos principales que los caracterizan, analizando fortalezas y debilidades desde la perspectiva de los usuarios. También revisaremos un poco la historia de estos sistemas, como se originaron y pasaremos por un conjunto de ejemplos de aplicaciones prototípicas que surgieron a partir de emprendimientos académicos, de grupos de investigación o de la industria. Esto se verá en las secciones de definiciones y conceptos básicos, comparación con las reuniones principales, un poco de historia, conceptualización de sistemas groupware y estado del arte.

Una vez clasificados los distintos tipos de sistemas colaborativos y luego de haber identificado los conceptos principales trabajaremos en la construcción de un lenguaje bien formado que permita expresar las características de los sistemas colaborativos para obtener modelos que se puedan ir refinando hasta obtener la implementación del sistema. Se mostrará en la sección de lenguaje de modelado de sistemas colaborativos (CSSL - Collaborative Software System Language) alternativas de sintaxis concretas con las que usuarios con distintos niveles de conocimiento podrán trabajar para especificar los sistemas que desean construir. También se mostrarán ejemplos para mostrar las distintas notaciones.

Luego explicaremos el método de especificación propuesto que guiará a los desarrolladores en las etapas iniciales del desarrollo. El método brinda una importante colaboración en el relevamiento y diseño de alto nivel de un sistema colaborativo. Asimismo sirve para establecer una documentación inicial para discutir con los potenciales usuarios el tipo de funcionalidad colaborativa que se pretende para el sistema.

Finalmente, puntualizaremos las conclusiones de este trabajo y describiremos los pasos a seguir vinculados con este trabajo donde se vislumbran los beneficios de contar con alguno de los resultados planteados.

4. Definiciones y conceptos básicos

Antes de comenzar con el desarrollo de este trabajo, brindaremos al lector algunas definiciones y conceptos básicos con el objeto de formar un vocabulario común que guíe la lectura del documento y, a la vez, contribuya a caracterizar los sistemas de groupware. Estos conceptos fueron rescatados de distintos trabajos científicos de la comunidad groupware y serán los elementos básicos con los que se trabajará en el método de especificación que se presentará oportunamente:

Groupware (o Sistemas Colaborativos) es una clase de aplicaciones, para grupos u organizaciones, que surge de la unión de computadoras, de grandes bases de información y de tecnologías de comunicación (esas aplicaciones pueden o no soportar cooperación). Es una tecnología diseñada para facilitar el trabajo en grupo. Se puede usar para comunicar, cooperar, coordinar, resolver problemas, competir o negociar.

Definimos groupware, entonces, como:

Sistemas basados en computadoras que soportan grupos de personas involucradas en una tarea común (u objetivo) y que proveen de una interfaz a un ambiente compartido **[1].**

Las nociones de una "tarea común" y "ambiente compartido" son cruciales en esta definición. A partir de ellas, se excluyen, por ejemplo, a los sistemas multiusuarios, donde los usuarios comparten algunos recursos – como espacio en disco, tiempo de procesamiento, memoria, etc. – pero no comparten en realidad una tarea común.

En un primer acercamiento a los sistemas groupware se pueden distinguir, por un lado, las aplicaciones donde los usuarios realizan actividades en forma simultánea (sincrónica)

llamadas “real time groupware”. Y por otro lado están aquellas en las que los usuarios realizan sus actividades en forma asincrónica y se denominan “non real time groupware.

A continuación veremos los conceptos más relevantes de los sistemas colaborativos.

4.1. Conceptos principales

Los conceptos fueron recopilados de distintos trabajos publicados y forman un vocabulario conocido por los miembros de la comunidad que trabaja con tecnología groupware.

Rol (CollaborationRole): Entendemos al rol como un conjunto de propiedades, conocimientos y responsabilidades que tendrá un usuario en un determinado momento. Permitirá entender cual será el papel que el usuario tendrá en todo momento cuando interactúe con otros usuarios en el sistema. Los usuarios podrán cambiar el rol dinámicamente y este cambio será manejado por los protocolos de colaboración. Por ejemplo, en algunas sesiones un usuario podrá actuar como “*mediador”*, en algunas situaciones o como “*participante”* en otras. [1], [2], [17]

Usuario (User): Representa a un usuario dentro del sistema. Contará con un conjunto de propiedades que lo identificarán. Por ejemplo un nombre, un nickname, su imagen y otros datos que identifiquen al usuario. . [1], [2], [17]

Objetos colaborativos (SharedObjects): Son los elementos creados por los propios usuarios que se pueden manipular en forma colaborativa gracias a las herramientas colaborativas que lo soportan. De tal manera que un usuario puede utilizar elementos creados por otros usuarios. Por ejemplo, en un ambiente de educación a distancia, un alumno puede editar elementos colocados por un compañero o el tutor también; cuando dos o más usuarios utilizan el chat, su conversación o chat log es un objeto colaborativo que puede ser usado por otras personas como material de discusión. La particularidad de estos objetos es que son editados por distintos usuarios y dependiendo de la herramienta y

del protocolo la edición puede ser en forma sincrónica (al mismo tiempo) o asincrónica. Los objetos colaborativos más comunes en los ambientes groupware son documentos de texto, dibujos, páginas web o chat_logs. [10], [17], [31]

Workspace (Workspace): Es el lugar en el que la *colaboración* se lleva a cabo y define, en parte, el estilo de colaboración que se va a llevar a cabo. En este trabajo se rescata a los workspaces como los elementos que contextualizan la colaboración. Por ejemplo, si pensamos en un aula virtual como un workspace, rápidamente entendemos qué tipo de actividades se van a realizar, qué roles intervienen, qué tipo de herramientas se van a usar y qué protocolos de colaboración intervienen. Para detallar este ejemplo podemos pensar aulas en las que se van a realizar actividades de clase al estilo tradicional y de consulta. En las clases tradicionales se usarán herramientas de video conferencia y foro y en las consultas se puede usar el mail y el chat. Los roles que intervendrán serán: docente, tutor y alumno. Finalmente podemos describir protocolos diferentes para la clase y para la consulta. En la clase la maestra podría coordinar la participación de los alumnos, dándoles la palabra en el caso que ellos la soliciten y en las consultas, los tutores podrían tener las mismas atribuciones que los alumnos al momento de participar en la colaboración. En definitiva, dentro de un workspace hay herramientas que son utilizadas para comunicarse y trabajar en los objetos compartidos. Los protocolos estructuran las interacciones de los roles en el workspace y el uso de las herramientas por ellos. En general un workspace o grupo de workspaces no son suficientes para *definir* una aplicación colaborativa. En el ejemplo que estamos siguiendo, vemos al aula como un workspace, pero probablemente haya otros espacios como la biblioteca o la cafetería que brindarán seguramente alternativas de colaboración distintas. Usualmente un workspace es definido dentro de un entorno más grande que lo contiene que llamaremos escenario colaborativo. [36], [47], [64], [65]

Sesión (Session): Una sesión es un período de interacción soportada por un sistema. En general un usuario ingresa (login) en la sesión identificándose a través de un nombre de usuario y password y sale de la misma (logout) en forma explícita o simplemente cerrando

la aplicación (o ventana) que maneja la sesión. La administración de la sesión entre otras cosas sirve para llevar un registro de las actividades de los usuarios mientras dura la sesión y generalmente se registra el día y la hora en que los usuarios se conectan a la misma.

Veamos por ejemplo una sesión en la que un usuario se conecta para bajar sus correos electrónicos usando un programa cliente de correo electrónico como el Outlook o el Eudora. El usuario se identifica a través de su nombre de usuario y password, el programa baja la información desde el programa servidor de correo electrónico hacia su computadora personal y finalmente se desconecta. La sesión en este caso, dura mientras el programa cliente está bajando los correos electrónicos. En este ejemplo el usuario no participa directamente en la sesión, sino que son los sistemas quienes intercambian información durante la sesión. Veamos otro ejemplo de sesión en la que un usuario escribe un documento o página web en forma remota. El usuario, mediante un programa corriendo en su computadora, participa en una sesión, la que permanecerá abierta mientras el usuario escribe el documento. En esté ejemplo el usuario participa activamente en el y transcurso de la sesión.

Cuando se estableces una sesión en un sistema groupware el usuario permanece un período de tiempo interactuando con el sistema, en forma similar al ejemplo anterior donde el usuario escribe un documento en un espacio remoto. Ese período de tiempo, puede superponerse con el periodo de tiempo de otro usuario. En este caso, si el sistema lo permite, esos usuarios tendrían la posibilidad de interactuar sincrónicamente (comunicarse, discutir, coordinar). [1]

Hacemos notar que desde el punto de vista de la implementación, la sesión se podría utilizar como un objeto para intercambiar información entre los usuarios conectados al sistema. Por ejemplo datos del usuario, nombre, rol, estado, etc. que se comunican para saber quien está conectado en ese momento en el sistema. En definitiva, la forma en que los sistemas groupware manejan la sesión es una de las principales diferencias con los sistemas monousuarios. [19], [13], [20]

Herramientas colaborativas (Tool): Son programas que, a diferencia de las aplicaciones monousuarios, contemplan ser utilizados por un grupo de personas. Naturalmente, el grupo manipula elementos que llamaremos objetos colaborativos que son los productos que se obtienen como resultado de la colaboración. Ejemplos de herramientas colaborativas son:

- Chat
- Pizarrón compartido o shared blackboard
- Editores colaborativos.
- Graphical tools

Estas herramientas pueden ser independientes del protocolo involucrado. Es decir que una misma herramienta puede ser utilizada con distintos protocolos. Por ejemplo el Chat podría ser usado de forma que los usuarios puedan enviar mensajes en cualquier momento o que la participación de los usuarios sea por turnos. En este ejemplo tenemos una misma herramienta usada en base a distintos protocolos. [41], [42], [40]

Escenario Colaborativo (Setting): Es la integración de un conjunto de workspaces. A menudo, las aplicaciones groupware complejas necesitan más de un workspace, de una manera similar a una universidad virtual que estará compuesta por distintos workspaces como el aula, la biblioteca, etc. Los escenarios contienen asimismo los protocolos que estructuran el acceso y el uso de los diferentes workspaces por parte de los distintos roles existentes. Por ejemplo podemos pensar en que algunos roles no podrán ingresar a algunos workspaces o que al cambiar de workspace el rol se convierte. Los escenarios colaborativos son un caso particular de Workspace que puede contener otros workspaces de manera tal de crear un ambiente más complejo de colaboración. Por ejemplo, podemos pensar en una Ciudad Colaborativa que contiene a una Universidad Virtual Colaborativa y a un Museo Colaborativo. [66]

Asociación (CollaborativeAssociation): Permite asociar elementos del diseño. Con esta asociación vinculamos las herramientas que disponemos en un workspace, los roles que pueden participar de una sesión etc. Existen diversas asociaciones entre los elementos estos últimos elementos que se expresarán en el método a través de Asociaciones colaborativas. Por ejemplo LocationRelationship que modelará la ubicación de objetos y roles dentro de los espacios, UseRelationship que modelará el uso por parte de los usuarios de los objetos colaborativos, ParticipationRelationship que modelará la participación de los usuarios en las sesiones, SessionRelationship que servirá para modelar la relación de las sesiones con los espacios. Todas ellas serán luego descriptas en el metamodelo. [1], [10]

Contexto compartido (SharedRepository): Es el repositorio donde se alojan los objetos colaborativos. Los usuarios tendrán acceso a este repositorio para crear, leer y modificar los objetos colaborativos y de acuerdo a la política de cada sistema colaborativo el contexto compartido puede ser único (es decir que todos los usuarios de la aplicación comparten a todos los objetos colaborativos en un lugar). En otros casos, puede haber varios contextos compartidos, asociados a alguna componente del sistema. Por ejemplo cada usuario podrá tener un espacio en el que coloca los documentos que quiere compartir. Asimismo, puede determinar quién puede acceder a su contexto compartido y qué acciones puede realizar dentro del mismo. En otros casos, los contextos compartidos pueden estar asociados a determinados espacios. Por ejemplo, cada aula podría tener un contexto compartido donde los usuarios comparten sus trabajos. Finalmente los contextos compartidos podrían generarse dinámicamente cuando un usuario decide compartir algún documento con otro usuario. En este caso el contexto compartido se generará en el momento y vivirá mientras los usuarios trabajen sobre el documento. Al igual que las sesiones, estos espacios de información compartida por los usuarios son una de las principales diferencias con los sistemas monousuarios y las acciones que los usuarios realizan sobre los documentos en estos espacios compartidos pueden ser visualizadas en el momento por los usuarios que están conectados a ese espacio compartido. [32], [43]

Telepuntero (Telepointer): Es el cursor del mouse de cada uno de los usuarios que está conectado a un objeto colaborativo y que puede ser movido por cada usuario. En algunos casos los cursores de los usuarios pueden ser visualizados todos al mismo tiempo. En otros casos, se ve solamente uno por vez, pero gracias al protocolo de colaboración los usuarios van cambiando el control sobre el cursor. Este recurso permite que un usuario señale alguna parte del objeto colaborativo en el transcurso de la colaboración. El movimiento del teleponinter se refleja en los monitores de los usuarios que están conectados al objeto colaborativo. [41]

Protocolo (Protocol): Un factor importante en el trabajo en grupo es el *proceso* social que se lleva a cabo. Sin gente interactuando, el sistema groupware está muerto. Los protocolos sirven para modelar, guiar y estructurar el proceso social que se lleva a cabo dentro del grupo. Los protocolos definen qué herramientas y objetos colaborativos pueden ser utilizados por los distintos roles o usuarios. Un aspecto importante que define un protocolo es en qué momento puede participar cada usuario. Por ejemplo, supongamos que dos usuarios están utilizando el Chat; la participación de los usuarios podría estar controlada por un protocolo en donde los usuarios escriban por turnos, similar a un debate televisivo. Es decir, el protocolo guía u ordena la participación de los usuarios y en herramientas más complejas, el protocolo puede determinar que operaciones dentro de dichas herramienta cada usuario puede usar en distintos momentos. Hacemos notar que el protocolo básico es el que permite que los participantes colaboren en cualquier momento y se lo denomina “no-protocol”. [48]

Vista (View): Es una porción del contexto compartido que puede ser visualizada por un usuario. En algunos casos, los usuarios pueden visualizar todo el contexto compartido, y en ese caso la vista es una representación del contexto compartido. En otros casos, por limitaciones de los dispositivos, la vista representa una porción de la información. La vista de cada usuario puede ser muy diferente, por ejemplo, podemos tener usuarios que cuenten con diferentes resoluciones en sus computadoras, incluso algunos usuarios pueden estar usando dispositivos móviles como celulares o computadoras palms. Por otro lado

algunos usuarios pueden estar mirando la misma información pero en diferentes representaciones (o configuraciones). Por ejemplo, un arreglo de números puede ser mostrado como una tabla en la vista de algún usuario o como un gráfico en otro. En definitiva un usuario podría tener una representación multimedial con video, sonido etc. y otros quizás tengan una vista solamente de texto. [1], [66], [57]

Meta protocolo (Meta-Protocol): En algunas ocasiones es necesario contemplar una forma de cambiar de protocolo. En síntesis el Meta protocolo es el protocolo que administra los cambios de protocolo. Como mencionamos antes, en un workspace se pueden desarrollar colaboraciones usando distintos protocolos. Como es el caso del Aula virtual que sirve para dar clase y para hacer consultas. La clase y las consultas tienen naturalmente distintos protocolos. Un posible Meta protocolo para un Aula Virtual puede ser el que determine que luego de una clase los alumnos tendrán a su disposición una consulta. Los meta protocolos controlan el cambio y la transición entre los protocolos y proveen una forma de hacer que los workspaces sean más flexibles. [48]

Acoplamiento: El acoplamiento es una medida que determina el grado de ensamble que tiene la misma componente en los puestos de trabajo de un sistema groupware. Las componentes pueden estar acopladas o no y el acoplamiento lo mencionaremos como un acoplamiento fuerte o débil. Si decimos que el sistema tiene la vista fuertemente acoplada, significa que los usuarios tienen la misma vista. Si tienen el workspace débilmente acoplado (también podemos usar el término desacoplado), estamos diciendo que los usuarios pueden estar en distintos workspaces. El mismo concepto puede aplicarse a otros conceptos como el protocolo, la herramienta, el telepointer, etc. Esta medida determina en cierta forma la complejidad del sistema y de la colaboración. En general el sistema va a ser más simple (mas simple para desarrollarlo y también mas simple para operarlo) cuando sus componentes estén fuertemente acopladas. Pero en algunos casos, un sistema débilmente acoplado permite mayores grados de acción a los usuarios y puede convertirse en un sistema más productivo. Si contamos con un sistema cuya vista está débilmente acoplada, el sistema deberá contar con la funcionalidad de cambiar la vista y si un usuario

cambia los objetos que está mirando el sistema, seguramente deberá notificarle a los otros usuarios, que objetos está mirando para poder colaborar con él. En caso contrario, si el sistema no permite tener vistas diferentes, quizás un usuario deba esperar (o negociar con el resto de los usuarios) a que se finalice la colaboración con los objetos que están mirando para cambiar la vista (y la del resto de los colaboradores) y poder continuar la colaboración con otra parte del sistema. Está claro que tener la vista desacoplada, permite que los usuarios trabajen en distintos espacios del contexto compartido, pero el sistema es más complejo para desarrollarlo y comprenderlo. En el contexto de nuestro trabajo, este concepto nos servirá para especificar características importantes del sistema y nos servirá para determinar la complejidad del mismo en el momento de desarrollarlo. [32]

Awareness: Es la información que el sistema provee sobre el estado de la colaboración. En las reuniones presenciales estar al tanto de los otros (staying aware of others) es algo natural. Se puede percibir dónde está ubicado cada uno, cuál es su estado, que actividad está desarrollando y con que objeto. Por el contrario, mantener actualizada esta información en sistemas groupware es bastante difícil. Es por esto que los primeros sistemas groupware, que no mantenían esta información de manera eficiente, resultaban confusos e ineficientes comparados con el trabajo cara a cara. El awareness del workspace involucra mantener constantemente actualizada la información de los otros usuarios en relación al espacio compartido indicando al menos la identidad y la presencia de los usuarios. Conjuntamente con esta información suele aparecer otra información de awareness como la actividad que están desarrollando, su ubicación dentro del sistema, su estado, que acciones va a desarrollar, que cambios está realizando, objetos que se utilizan, etc. Esta información facilita el trabajo de los usuario, permitiéndoles tener una mejor percepción o conciencia de lo que está pasando con cada uno de ellos y con los otros usuarios en el ambiente colaborativo. [50], [58]

Avatar: Es la representación de un usuario dentro del sistema. Puede ser una pequeña imagen, un gráfico o un ícono. En algunos casos, como en los ambientes virtuales, el avatar se mueve dentro del ambiente y sirve para iniciar colaboraciones con ese usuario.

El avatar puede servir como información básica de awareness y en muchos sistemas colaborativos indican la presencia y la ubicación de los usuarios dentro de workspaces. [50], [58]

La recopilación de estos conceptos es útil ya que conforman un vocabulario común que facilitan el entendimiento entre los desarrolladores o entre los desarrolladores y los potenciales usuarios de sistemas colaborativos. De esta forma a todos les queda claro a que nos referimos cuando hablamos de alguno de estos conceptos. Por otro lado, serán de utilidad a la hora de describir alguna característica de algún sistema groupware.

5. Comparación con las reuniones presenciales

En este capítulo analizaremos las fortalezas y debilidades de las sesiones en los ambientes colaborativos llamadas real time groupware desde la perspectiva de un usuario comparándolas con las reuniones presenciales [39]. Estos sistemas se caracterizan por tener sesiones donde los usuarios colaboran entre sí en forma sincrónica y distribuida que llamaremos sesiones distribuidas.

Realizaremos una primera aproximación analizando en qué situaciones las sesiones distribuidas tienen ventajas sobre las sesiones cara a cara y en cuáles presentan desventajas. Dejaremos fuera de este análisis la ventaja trivial que es la que indica que los usuarios de sesiones distribuidas no necesitan trasladarse para participar de una sesión. Esta ventaja será seguramente analizada en el plano económico de las organizaciones, cuando evalúen costos y beneficios de contar con reuniones virtuales o presenciales. Discutiremos en esta sección, cuándo es conveniente realizar una sesión presencial y cuándo una sesión cara a cara, suponiendo que podemos elegir entre estas dos alternativas.

Entre las ventajas de las sesiones distribuidas se encuentran:

- ***Incrementan el acceso a la información:*** los participantes de una sesión distribuida tienen acceso a sus archivos, residiendo cada uno en su oficina, lo que les permite acceder fácilmente a información que puede ser relevante y, que de otra manera, podría no estar disponible durante una sesión. Muchas veces algunas reuniones presenciales tienen que ser suspendidas o postergadas por falta de información. Finalmente, los usuarios ganan en comodidad y familiaridad al mantenerse en sus oficinas.
- ***Estimula el trabajo paralelo dentro del grupo:*** Es sabido que el trabajo grupal permite reunir múltiples perspectivas y habilidades de los distintos participantes y que mejora

los resultados obtenidos. En general el trabajo en grupo permite dividir las tareas y armar subgrupos que se enfoquen en cada tarea y luego los trabajos de cada subgrupo son combinados para obtener el resultado final. Esto permite que las tareas de cada subgrupo trabaje en forma paralela ganado tiempo en obtener el resultado esperado. En el caso de las sesiones distribuidas, la división de tareas y la asignación de las mismas a los grupos y subgrupos es natural y consecuentemente se obtiene un porcentaje más alto de trabajo en paralelo. Asimismo, las sesiones distribuidas brindan otras posibilidades de realizar trabajos en paralelo. Es común para usuarios en un grupo distribuido ausentarse por un momento (para trabajar sobre otra pantalla, leer el correo electrónico, buscar información, etc.) y luego reanudar la sesión para continuar colaborando. Esto no es socialmente aceptable en un trabajo grupal "cara a cara", pero es aceptado en sesiones distribuidas. También es posible, participar en más de una sesión al mismo tiempo. En general los sistemas colaborativos permiten que el usuario mantenga, por ejemplo, una sesión de Chat con un grupo de usuarios, mientras escribe un documento con otros. Naturalmente esto no es posible en las sesiones cara a cara ya que una persona no puede estar en dos lugares al mismo tiempo.

- ***Permite el trabajo anónimo:*** En algunas situaciones los participantes pueden sentirse más relajados al expresar ideas en un grupo si lo hacen en forma anónima. En otros casos es imprescindible que los usuarios no se identifiquen, por ejemplo en votaciones electrónicas, en el momento de participar en una sesión colaborativa.
- ***Garantiza resultados de la colaboración:*** Las sesiones distribuidas generan necesariamente algún resultado concreto. Un Chat entre dos usuarios, que es la mínima expresión de sesión distribuida, genera al menos como resultado el texto de la conversación de los usuarios. En reuniones cara a cara, muchas veces los participantes salen de las mismas sin llevarse un resultado concreto de las mismas.

Entre las desventajas de las sesiones distribuidas están:

- ***Se generan discusiones incomprensibles:*** Las sesiones distribuidas tienen un patrón de comunicación marcadamente diferente a las sesiones "cara a cara". Por un lado, el medio que más se usa para transmitir información es el texto. En consecuencia, muchas de las expresiones o gestos que las personas usamos en reuniones cara a cara tenemos que expresarlas en formato de texto y pueden no ser adecuadamente entendidos por los receptores de la información. Por otro lado, cuando hay muchos participantes colaborando al mismo tiempo (como suele suceder en los chats masivos) y las herramientas no soportan protocolos de colaboración que organicen la discusión, es muy difícil seguir el hilo de la discusión. En los casos donde se permite transmitir información por otros medios como el sonido, en la mayoría de los casos la comunicación no es full-duplex, sólo la voz de una persona es transmitida por vez, consecuentemente los sistemas tienen que manejar eficientemente la coordinación para que la participación de los usuarios sea efectiva. Finalmente, los sistemas colaborativos no pueden transmitir toda la información que se percibe en las reuniones cara a cara. En muchos sistemas donde la información de awareness es deficiente, los usuarios no comprenden la información que brindan el resto de los usuarios, porque no conocen información relevante del contexto como ser, qué objeto está manipulando, qué estado tiene el usuario, qué herramienta está usando, etc.
- ***Requiere más concentración para el trabajo en grupo:*** Como consecuencia del punto anterior, los usuarios requieren mayor concentración para comprender lo que se está discutiendo. Frecuentemente las sesiones "cara a cara" parecen ser más cortas, y suelen ser más alegres. En cambio las sesiones distribuidas son más formales y pueden resultar más cansadoras. Por otro lado, las deficiencias en información de awareness que pudiera tener el sistema tiene que ser cubierto por cada usuario para intuir o suponer qué objetos está manipulando el usuario, cuál es su rol, etc.
- ***Reduce la interacción social:*** Las sesiones distribuidas tienden a ser más serias, los usuarios ponen el foco de atención sobre la tarea específica y en general no surgen temas extra laborales. Esto genera una ganancia en eficiencia y tiempo, pero reduce el

intercambio social. Muchas de las sesiones “cara a cara” parecieran ser más intensas, con interacciones más ricas. Si bien raramente los miembros del grupo se miran directamente entre ellos, el hecho de estar en la misma habitación se incrementa la información de contexto sobre los otros miembros y puede dar pie a conversaciones que vayan más allá de los temas específicos de la reunión.

Vimos las características de las sesiones sincrónicas en relación a las reuniones presenciales cara a cara. En la comparación se analizaron ventajas y desventajas en cada caso con resultados positivos y negativos. De todas formas entendemos ambas (las presenciales y las remotas) pueden ser complementarias. En muchos casos se utilizan las sesiones virtuales para preparar material para alguna otra reunión principal; por ejemplo eligiendo los temas que se van a tratar o el orden en que van a hacerlo. También puede darse el caso inverso, en el que una reunión presencial dispara diferentes líneas de acción que podrán ser documentadas en reuniones virtuales con herramientas que estimulan la escritura colaborativa; por ejemplo una wiki.

En definitiva vemos que cada vez va a ser más común conectarse con otras personas a través de sistemas que soporten la comunicación y la colaboración. Entender las diferencias con las reuniones presenciales servirá para potenciar las ventajas y minimizar las desventajas. Por otro lado, la integración de los dos casos o la combinación puede ser de utilidad para mejorar la productividad del trabajo en grupo.

6. Conceptualización de sistemas groupware

En esta sección presentaremos distintos enfoques conceptuales que permiten analizar aquellos aspectos de las aplicaciones groupware que son significativos para la toma de decisiones sobre los ambientes colaborativos más adecuados a las distintas necesidades de trabajo que pueden tener empresas o usuarios individualmente.

Buscaremos conceptualizar distintos aspectos de las aplicaciones groupware. Hay diferentes enfoques para analizar que tipo de ambiente colaborativo la empresa (o el usuario) está necesitando.

Uno de los enfoques consiste en analizar tres aspectos claves para soportar una eficaz interacción de un grupo de trabajo: *comunicación*, *colaboración*, y *coordinación* [1].

6.1. Comunicación, colaboración y coordinación

Como mencionamos en la definición, los sistemas groupware, soportan a un grupo de personas involucradas en una tarea común en un ambiente compartido. La colaboración entre los participantes del grupo permitirá llegar al objetivo o tarea que desean realizar. Una efectiva *colaboración* requiere que los usuarios del sistema groupware compartan información y que puedan trabajar sobre ella. Para ello necesitarán acceder a un contexto compartido en el que puedan trabajar sobre los objetos compartidos. El contexto deberá estar actualizado y brindando notificaciones explícitas de las acciones de cada usuario.

Pero el trabajo sobre los objetos compartidos no alcanza para realizar una colaboración efectiva. Los usuarios requieren herramientas para comunicarse, a través de las cuales puedan discutir, negociar o intercambiar opiniones. Las herramientas de comunicación pueden ser sincrónicas o asincrónicas y pueden estar basadas en texto como el Chat o el

Correo electrónico o transmitir diferentes medios como sonido o video, como por ejemplo, los sistemas de video conferencia o los sistemas de Voz sobre Ip. Es necesario destacar que estas herramientas son las que más se han desarrollado; pero, usualmente, trabajan en forma independiente y no se integran con otras herramientas que soporten la colaboración o la coordinación. Por otro lado, hay otras formas de comunicación como la telefonía que van integrándose a la comunicación a través de computadoras. En este sentido tenemos los casos de mensajes de texto que pueden enviarse a través de teléfonos celulares o sistemas de computadoras que permiten realizar llamadas telefónicas a teléfonos fijos o celulares. La integración de las telecomunicaciones y de las tecnologías de procesamiento de datos ayudará a disminuir las diferencias entre la comunicación tal cual la conocemos hoy con la comunicación mediada por la computadora.

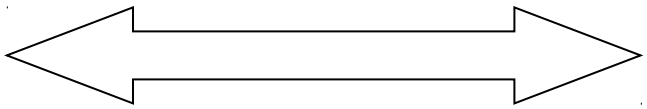
La efectividad de la comunicación y colaboración puede ser incrementada si las actividades del grupo están coordinadas. La *coordinación* permite organizar la participación de los usuarios en el proceso colaborativo. A partir de ella se determinan las actividades que puede hacer cada usuario y en que momento las pueden hacer. Por ejemplo el caso de VITAL [67], que es un ambiente para aprendizaje colaborativo, preparado para soportar grupos pequeños y medianos de usuarios, existe un elemento de coordinación en el auditorio cuando el rol de entrenador (Trainer) elige que la sesión sea guiada o cooperativa. En definitiva la coordinación permite que los integrantes no entren en conflictos al intentar acceder a un recurso compartido. Un equipo de redactores trabajando en una sesión sin coordinación, podrían entrar en conflicto o repetir ciertas acciones realizadas por otros. Siguiendo con este ejemplo los redactores van a necesitar usar los espacios de comunicación para avisar al resto de los usuarios cuales son los recursos que está usando cada uno. En definitiva, los sistemas colaborativos que tienen buenos elementos de coordinación simplifican la comunicación. Por otro lado, la colaboración se ve beneficiada, por que se requiere menos esfuerzo para decidir quien puede hacer que cosa en que momento.

6.2. Espectro de sistemas colaborativos

Los sistemas que soportan tareas comunes y ambientes compartidos varían ampliamente; es, por tanto, apropiado pensarlos más en términos de un espectro con múltiples dimensiones, en el que los distintos sistemas se ubican en diferentes puntos del mismo. En esta sección describiremos las dimensiones que nos permiten precisar las diferencias entre los sistemas de una manera exhaustiva y estable en el tiempo.

6.2.1. Según su nivel de integración:

Se describen aquí dos sub-dimensiones, una de acuerdo al grado de integración que tienen los usuarios en la tarea que realizan; y la otra en relación al grado de integración de los usuarios en el ambiente compartido. Asimismo se mencionan un par de ejemplos de aplicaciones que se ubicarán en los extremos de cada sub-dimensión

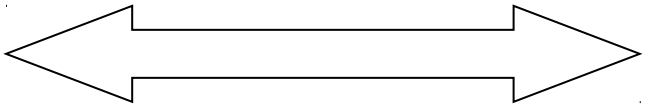


6.2.1.1. Sub-Dimensión de Tarea Común

Bajo:	**Alto:**
Los usuarios realizan tareas en forma independiente. Cada usuario utiliza algún recurso del sistema pero el sistema no fomenta la integración de los usuarios. En el mínimo nivel de integración se encuentra los sistemas de Tiempo compartido	Los usuarios realizan sus tareas en forma conjunta, dependen unos de otros y hay una constante percepción de las actividades que realizan los demás colaboradores. En el máximo nivel de integración se encuentran los sistemas de escritura colaborativa.

Los sistemas de tiempo compartido, atienden a un conjunto de usuarios que realizan tareas independientes entre sí. Cada usuario utiliza recursos del sistema para realizar su tarea. Por eso se considera bajo el nivel de integración desde el punto de vista de la realización de una tarea común. Existen algunas organizaciones que cuentan con grandes computadoras que pueden ser usadas en forma remota por un grupo de usuarios para realizar trabajos específicos, procesamiento de datos o cuentas de alta precisión o complejidad. Este escenario aporta un ambiente donde la integración de las tareas de los usuarios es mínima.

Por otro lado, los sistemas de edición colaborativa, permiten a un conjunto de usuarios escribir un documento en forma conjunta. En este caso, el grupo de usuarios está realizando una tarea con un alto nivel de integración desde el punto de vista de la tarea a realizar.



6.2.1.2. Sub-Dimensión de Ambiente Compartido

Bajo:	**Alto:**
Los usuarios colaboran en ambientes independientes. Por ejemplo los sistemas de correo electrónico	Los usuarios colaboran en un ambiente compartido donde comparten información y se comunican como es el caso de los sistemas de aulas virtuales.

La dimensión que clasifica a los sistemas de acuerdo al nivel de utilización del ambiente nos muestra en un extremo a los sistemas de mail o Electronic Mail System. Este sistema

permite enviar y recibir correos electrónicos y cada usuario tiene su ambiente de trabajo para hacerlo. Por ello es que tiene un bajo nivel de utilización del ambiente compartido. En el otro extremo encontramos, por ejemplo, a los sistemas de aulas virtuales, donde los usuarios trabajan en un ambiente compartido. Estos sistemas permiten a un conjunto de usuarios con distintos roles (instructores, profesores, alumnos), emulando un aula tradicional, presentar material, discutir algún tema, preguntar y responder. Todo esto se lleva a cabo en un ambiente de alta integración ya que los usuarios, comparten herramientas e información. Por ejemplo el material de clase o la lista de usuarios que están en la misma.

6.2.2. Según el tiempo y el espacio

Los sistemas groupware pueden ser concebidos tanto para ayudar a grupos en los cuales las personas se encuentren en un mismo lugar, como para grupos en los cuales sus integrantes estén distribuidos en distintas ubicaciones. A su vez, un sistema de groupware puede ser concebido para reforzar la comunicación y la colaboración dentro de una interacción en tiempo real (sincrónica), o de una interacción que no se lleva a cabo en tiempo real (asincrónica).

Estas consideraciones de tiempo y espacio sugieren cuatro categorías de sistemas de groupware representados en el siguiente cuadro de doble entrada:

6.2.2.1. Según el tiempo y el espacio:

	Mismo Tiempo	**Diferente Tiempo**
Mismo Lugar	*Interacción cara a cara*	*Interacción asincrónica*
Diferente Lugar	*Interacción distribuida Sincrónica*	*Interacción distribuida Asincrónica*

Veamos los ejemplos de sistemas que se ubican en los diferentes cuadrantes de la tabla anterior. La tecnología que se utiliza para soportar las reuniones llamadas tecnologías "meeting room" se ubica dentro del cuadrante izquierdo superior; es decir, mismo lugar-mismo tiempo: este tipo de sistemas consiste de una habitación con una gran pantalla común conectada a un conjunto de computadoras interconectadas. Estos sistemas han surgido con el objetivo de mejorar los encuentros cara a cara y sirven para procesos como *brainstorming*, organización y evaluación de ideas.

La tecnología de shared whiteboard o pizarrón compartido se encuentra dentro de la celda inferior izquierda, este tipo de sistemas fueron diseñados para dar soporte a encuentros donde se hacen discusiones acerca de diseño, por ejemplo. En este tipo de encuentros las personas dibujan sus diseños y apuntan ítems particulares y relaciones. Cada persona del encuentro puede referirse a los dibujos y proponer modificaciones alterándolos. Los objetos dibujados en el pizarrón compartido están visibles a todos los usuarios.

En el cuadrante superior derecho podemos ubicar a la tecnología de bulletin boards o news groups, este tipo de sistemas son una variante de los sistemas de mail. En lugar de enviar un mensaje a través de una computadora a una o más personas (como es el caso del mail), estos sistemas permiten a los usuarios enviar un mensaje a un "lugar" identificado unívocamente en el espacio utilizado para discutir acerca de un tema en particular.

Por último, el sistema de correo electrónico en la celda inferior derecha, es el ejemplo más familiar de groupware. Permite intercambiar mensajes textuales en forma asincrónica. Otro ejemplo de sistemas que se pueden ubicar en esta celda son los sistemas de group schedulling. Programar un encuentro con un grupo de personas es una de las tareas de grupo que podría beneficiarse con el soporte de computadoras. Encontrar tiempo libre en varias agendas personales es una tarea que puede ser hecha más eficientemente mediante un sistema de groupware. Hay ejemplos de estos sistemas que utilizan un mecanismo de voto para organizar el plan de encuentro. Otros sistemas más nuevos involucran el

intercambio de una serie de mails que son invitaciones, respuestas de confirmación y negaciones.

Un sistema de groupware podría abarcar todos los cuadrantes. Por ejemplo, podría ser útil tener la misma funcionalidad base y el mismo look and feel de la interface de usuario (a) mientras estoy utilizando una computadora para editar un documento en tiempo real con un grupo (mismo tiempo / mismo lugar o mismo tiempo / diferente lugar) y (b) mientras estoy solo editando en mi oficina (diferente tiempo).

6.2.3. Restrictivo contra Permisivo

Otra dimensión del espectro que estamos analizando es el grado de libertad que tienen los usuarios para desarrollar sus tareas. Algunos sistemas colaborativos como los sistemas de workflows, restringen o dirigen el comportamiento de los usuarios. Este tipo de sistemas son considerados restrictivos y típicamente mantienen un curso de acción posible o deseable que restringe las posibilidades de acción de los usuarios. Estos sistemas son habituales en empresas comerciales para organizar y optimizar los circuitos de trabajo; los usuarios hacen lo que el sistema les indica realizar. Estos sistemas pueden o no tener instancias de colaboración sincrónica entre los usuarios.

Otros sistemas, tales como los sistemas de pizarrones compartidos o sistemas de escritura colaborativa permiten que los usuarios puedan libremente realizar sus actividades (escribir, navegar, dibujar, entrar o salir, etc.) sin control del sistema. Estos sistemas son habituales en ámbitos de construcción de conocimiento (encuentros de usuarios para diseño o investigación). Estos sistemas suelen tener espacios de colaboración sincrónica donde los usuarios se encuentran para compartir sus conocimientos [64], [65].

Los sistemas permisivos son más complejos de desarrollar. Por un lado, requieren soportar las alternativas de acción de los usuarios brindando la funcionalidad que corresponda (por ejemplo navegar entre los documentos, escribir mientras otros realizan otras actividades, etc.). Y por otro, requieren elementos de awareness que informen qué actividades los

usuarios están realizando, dónde la están llevando a cabo, con qué elementos, etc. Como ejemplo simple veamos el caso en que un grupo de usuarios se reúne para escribir conjuntamente un documento. Si el sistema es permisivo, permitiendo que los usuarios cambien la página que están visualizando del documento, el sistema deberá informar que parte del documento cada usuario está visualizando. Esa es una información de awareness que un sistema restrictivo no tiene la necesidad de implementar.

Analizamos diferentes enfoques para entender mejor las características de los sistemas colaborativos. Estos pueden usarse complementariamente para tener una visión más completa del tipo de funcionalidad colaborativa contamos. Estas clasificaciones nos permitirán entender rápidamente que si queremos trabajar con muchos usuarios lo más recomendable sería promover sesiones asincrónicas con funcionalidad que permita coordinar la participación de los usuarios en el contexto compartido. Asimismo, el sistema debería ser permisivo a la hora de elegir sobre que objetos trabajar o dicho de otra forma, no será obligatorio que todos trabajen sobre el mismo recurso compartido. En cambio si queremos trabajar con pocos usuarios podemos tener interacciones sincrónicas accediendo al mismo objeto del contexto compartido incluso con posibilidad de tener edición simultánea sobre el mismo objeto.

En definitiva estas clasificaciones, que combinan aspectos tecnológicos y de interacción entre personas, se aplican a un amplio rango de aplicaciones con diferentes niveles de complejidad tecnológica. [27]

7. Justificación del modelo espacial

El enfoque que presentaremos en esta sección nos permitirá comprender las aplicaciones groupware de una manera mas integral desde el punto de vista de cómo plantear estrategias de construcción de soluciones colaborativas. Asimismo nos permitirá pensar los estilos de colaboración y sus especificidades en términos de espacios. Es decir, traduciendo el conjunto de soluciones específicas a diseños particulares de espacios virtuales, sus vínculos mutuos, de la ubicación de los objetos, de la dinámica de la colaboración (protocolo de colaboración) y como van a interactuar los usuarios entre si y con los objetos, de la forma en que los usuarios ingresan, permanecen y salen del espacio. Esta forma de abordar el diseño de soluciones colaborativas tiene, entre otras ventajas, que desde el inicio el trabajo de elaboración está guiado, por la forma en que dichas soluciones ayudarán a los usuarios en una efectiva colaboración; poniéndolo a este, no como receptor final, sino como el fundamento de todo el proceso de diseño. De esta manera debería conseguirse el objetivo de que el resultado final este mucho más en sintonía con las necesidades y capacidades operativas de los usuarios reales y del grupo de usuarios.

Por otro lado el enfoque espacial nos permitirá contextualizar las características en una primera etapa del modelo de especificación que proponemos en este trabajo.

Para analizar las posibles características de los espacios virtuales, trazaremos comparaciones con algunas características de los espacios físicos. Luego veremos como deberían ser soportados estos espacios en los ambientes colaborativos.

En el mundo real la gente cambia continuamente entre diferentes estilos de colaboración en diferentes tiempos, diferentes lugares, formales e informales, en ambientes restrictivos o permisivos etc. En el caso de los sistemas groupware, muchas tecnologías fueron concebidas para mantener una situación colaborativa específica. Por ejemplo los

programas para comunicarse como los Chat o mensajeros instantáneos, o aquellos otros para coordinar actividades como agendas colaborativas, etc... En consecuencia, cuando una persona requiere cambiar de estilo de colaboración, tiene que cambiar de aplicación (en algunos casos tiene que cambiar de proveedor de software y en algunos otros casos tiene que ejecutar un entorno particular). Aquí coincidimos con [64] en que esto introduce una barrera (o *gap*) que interfiere en el desarrollo normal del proceso de colaboración. Más específicamente el gap se define como la barrera física o de percepción dentro del ambiente groupware que distrae o bloquea la forma en que los usuarios trabajan o desean trabajar.

Aparecen numerosas barreras en los sistemas groupware que requieren atención de los diseñadores. Entre ellas encontramos:

- El gap entre el trabajo individual y el trabajo grupal. El trabajo grupal introduce reglas propias algo más complejas que las del trabajo individual. Los protagonistas de la colaboración deberán esperar su turno para poder colaborar, compartir recursos, comunicarse y coordinar actividades.

- El gap entre el deseo de colaborar y el establecimiento de una sesión colaborativa real. Las diferentes realidades de los usuarios involucrados en una sesión de colaboración real puede marcar diferencias en el momento de establecerse la sesión. Por ejemplo, diferencias de horarios, sistemas de base, de ancho de banda, etc. puede hacer que los usuarios tengan problemas para colaborar.

- Existe una barrera tecnológica cuando se pasa de un software convencional a un software colaborativo. En general, los sistemas groupware son más complejos de instalar, difíciles de configurar y de poner en funcionamiento. A menudo los sistemas groupware no están integrados y se requiere trabajar en ambientes heterogéneos.

- Los diseñadores deberán atender las dificultades que los usuarios tienen para entender las diferentes fases de las actividades colaborativas. En efecto, los usuarios deberán trabajar con diferentes herramientas para completar sus tareas; por ejemplo, cuando los usuarios, utilizando algún software para registrar el aporte de nuevas ideas, realizan actividades previas a las reuniones, cambian de fase y pasan a utilizar otro sistema que permita discutir durante una reunión.

- La gap entre el mismo lugar y diferentes lugares, cuando un usuario quiere reunir a parte del grupo que está en la misma sala con otros usuarios que están en otros lugares. Los mecanismos para coordinar la colaboración deberán contemplar esta situación. También la información de awareness que brinde el sistema deberá ser diferente para los usuarios que están en el mismo lugar que para los que están distantes. Por ejemplo podemos tener un grupo de gerentes de una empresa trabajando con otro grupo de gerentes de otra empresa. La información de awareness de cada grupo mostrará principalmente información del otro grupo ya que no se requiere esa información para los que están en la misma sala.

- El gap entre reuniones formales e informales. Los diseñadores deberán poder transmitir las diferencias que hay entre las actividades formales e informales. En algunos casos las reuniones formales registrarán las actividades de los usuarios, por ejemplo en un sistema de educación a distancia, el sistema puede registrar cuántas veces los alumnos se conectaron al sistema, cuantas preguntas realizaron y cuáles documentos enviaron, etc. Por otro lado las reuniones informales pueden no tener mayores restricciones desde el punto de vista de los elementos a registrar. En todo caso los sistemas groupware deberán transmitirle al usuario las características de la reunión para que el usuario esté al tanto de cómo trabajar correctamente con el sistema.

- El gap entre herramientas electrónicas con otras de escritorio que no pueden ínter operarse

Trabajar con un modelo espacial, puede alimentar una amplia gama de estilos de colaboración diferente en el mismo sistema colaborativo, por ejemplo podemos tener espacios en los que los usuarios pueden colaborar sincrónicamente y en otras pueden hacerlo en distinto tiempo. Usar una metáfora de ambiente, "room methaphor", ayuda también a mitigar o remover algunos gaps técnicos y ayuda al usuario a entender conceptualmente la aplicación. Asimismo facilita la transición (puede ser organizada con links entre los diferentes ambientes) entre los diferentes estilos de colaboración.

El diseño de espacios físicos y como la gente los usa, es un campo bien conocido en áreas como planificación urbana y arquitectura. Similarmente, en los sistemas navegables hay varias investigaciones y metodologías, como OOHDM [26], que nos ayudan a construir ambientes virtuales que nos permiten recorrerlos fácilmente. En esta sección no enfocaremos estos métodos, sino que estudiaremos cómo vincularnos con ellos para diseñar sistemas colaborativos que traspasen más fácilmente las barreras mencionadas.

Particularmente, la metáfora de ambiente modela fácilmente qué es lo que la gente puede hacer en ese ambiente y es una forma natural de proveer oportunidades colaborativas. Los ambientes virtuales, análogamente a los ambientes físicos, pueden ser usados por los grupos dentro de la organización para realizar las actividades conjuntas. Presentaremos, a modo de ejemplo, las características de algunos ambientes que determinan el estilo de colaboración que se podría desarrollar. Entre ellos podemos encontrar:

- Oficinas Personales: Son espacios donde podemos ubicar a algún usuario con quien se desea colaborar. En estos espacios los visitantes tendrán menos derechos sobre los objetos compartidos que el dueño de la oficina. En general la cantidad de participantes es reducido (entre 3 o 4) y se puede contar con herramientas de edición colaborativa donde el dueño de la oficina coordinará la dinámica de la colaboración. Considerando este espacio, implícitamente estará determinado el protocolo de colaboración (qué cosa pueden hacer los usuarios y en que momento)

- Aulas Virtuales: Las aulas son espacios compartidos por un grupo compuesto por algunos roles bien determinados. En general se encuentra el docente o tutor y los alumnos. También existe un conjunto de herramientas colaborativas determinadas como pizarrones compartidos, repositorios donde los docentes colocan sus materiales para compartirlos con los alumnos y herramientas de comunicación como mensajeros instantáneos o foros para que los miembros del grupo se comuniquen. En este espacio se pueden establecer algunas alternativas de colaboración diferente, por ejemplo podemos contar con sesiones de clase (donde el docente presentará alguna información) o con sesiones de consulta donde los alumnos podrán comunicarse con el docente o tutor para consultar alguna duda que pudieran tener. En definitiva lo que vemos es que el mismo espacio (el aula virtual) puede ser utilizado con protocolos distintos para realizar actividades colaborativas diferentes.

- Salas de Reuniones: Son espacios pensados para que un grupo relativamente pequeño (entre 3 y 6 personas) se reúnan a tratar algún tema en particular. Existen diversos protocolos que guían las reuniones, por ejemplo podemos reunirnos a intercambiar ideas bajo el protocolo conocido como brainstorming (tormenta de ideas) o podemos tener una reunión con el protocolo de debate en el que los participantes colaboran por turnos. Además de herramientas convencionales de comunicación (foros o chats) o edición colaborativas (como pizarrones compartidos) que pueden aparecer en otros espacios, pueden aparecer herramientas específicas para soportar algunas dinámicas como las de brainstorming. Otra característica interesante es la posibilidad de cambiar dinámicamente de protocolo. Por ejemplo en una reunión virtual podemos primero rescatar algunas ideas con el protocolo de brainstorming y luego debatir las ideas con el protocolo de debate. En estos casos suele haber algún rol (en general es el coordinador de la reunión) el que podrá cambiar de modalidad en el momento que lo crea conveniente.

- Cafetería: Son espacios pensados para que los usuarios del ambiente se encuentren en comunicaciones informales. En general cuentan con diversas herramientas de comunicación de distinto tipo. Los mensajeros instantáneos para comunicaciones on-line, o carteleras de mensajes para colocar mensajes que serán leídos por otros usuarios cuando accedan a la cafetería. En estos espacios el protocolo es básico y los usuarios participan en el momento que quieran en las distintas sesiones que se realicen.

Describimos a continuación una tabla con las características de los espacios en relación al sistema en general (ambiente colaborativo) y un comentario con el comportamiento de estos ambientes que deberán ser soportados en los sistemas colaborativos

Característica	**Como debería ser soportado en los sistemas colaborativo**
Escenario Colaborativo está compuesto por un conjunto de espacios.	Los espacios son representados dentro de ventanas. Como es sabido un usuario podría abrir mas de una ventana a la vez, en consecuencia, los usuarios podrían estar en más de un espacio a la vez.
Los espacios contienen elementos.	Los espacios contienen herramientas colaborativas (de comunicación como foros y chats o de trabajo como pizarrones compartidos o editores colaborativos) usuarios y objetos compartidos. Las cantidades de usuarios y objetos recomendables dependerán de las características de las herramientas colaborativas y del protocolo de colaboración. Por ejemplo, si tenemos una herramienta para enviar mensajes a un foro, podremos tener muchos usuarios participando en la colaboración. Por el

	contrario, si tenemos una herramienta sincrónica, como un editor colaborativo, es probable que el sistema funcione aceptablemente con menos de diez usuarios.
Permeabilidad Los usuarios y objetos colaborativos pueden entrar y salir de los espacios.	La gente puede navegar entre distintos espacios. Los vínculos entre los rooms pueden estar en cualquier lugar. Sin embargo en los ambientes colaborativos es mas difícil mostrar que es lo que está pasando dentro de un room y depende de la funcionalidad de awareness que tenga el sistema. Asimismo los objetos colaborativos se pueden ingresar o sacar del espacio. El manejo de objetos dentro de los rooms puede ser similar a los ambientes reales, pero usualmente se copian los ítems desde y hacia otros rooms.
Persistencia. Los objetos que se encuentran dentro de los espacios deberán persistir para la próxima colaboración	Los espacios mantienen el estado a través del tiempo. Esto incluye el contenido de las herramientas (objetos colaborativos). Incluso cuando se apaga y se prende el servidor. Existen algunas restricciones, en función de privacidad y de los roles habilitados a ver algunos contenidos o utilizar herramientas.
Personalización.	Los usuarios configuran sus ambientes dando controles de acceso a sus objetos compartidos para el resto de los usuarios (definiendo cual de ellos es privado, público o compartido con algún subconjunto de usuarios). También pueden especificar en que lugar del ambiente se ubican las cosas. En algunos sistemas colaborativos cada usuario puede realizar algunas configuraciones de personalización (colores, tamaños y ubicación espacial de algunos objetos) en todos los ambientes aunque no sean suyos. Estas personalizaciones pueden convivir con las de

	los otros usuarios. Es decir que dos usuarios podrían estar mirando el mismo ambiente, con características personales. Estas características de personalización deberían almacenarse y cargarse cada vez que el usuario se conecta al sistema.
Propiedad sobre los objetos y privacidad	Generalmente los espacios tienen nombre y poseen un propietario. Las reglas de control de acceso sobre los objetos compartidos de los espacios son definidas implícita o explícitamente (por el propietario del espacio).
Relación espacial y orientación	La ubicación de los objetos en el espacio facilita la orientación de los usuarios dentro del ambiente. En muchas aplicaciones colaborativas el usuario solo aparece en una lista de usuarios presentes (es decir que no tiene una representación en el espacio) y no ocupa un lugar en particular. Lo mismo puede pasar con algunos objetos y aplicaciones que pueden ser seleccionados de una lista.
Proximidad, acción y reciprocidad	Cuando el usuario tiene una representación en el espacio, usualmente se lo representa con una imagen que lo representa "avatar". Las acciones que realizan los usuarios es indicada con elementos de awareness como el "telepointer" o informando las acciones de los usuarios en la lista de usuarios. Algunas acciones de los usuarios pueden depender de la proximidad con otros usuarios (por ejemplo dos usuarios pueden iniciar una comunicación cuando estén lo suficientemente cerca uno del otro) o de la proximidad con otros objetos, por ejemplo un usuario podrá ver el contenido de un objeto si se encuentra cerca de él. Para que los usuarios tengan esta misma percepción el sistema tiene que mantener información sobre las acciones de los usuarios y los objetos que están manipulando. Esto hace una colaboración

	más efectiva.

A continuación describiremos el comportamiento de los sistemas colaborativos en relación a los usuarios del sistema.

Característica	**Como deberá ser soportado en los sistemas colaborativos**
Presencia. Cuando los usuarios ingresan al sistema (o al espacio o a la herramienta colaborativa) se deberá indicar la presencia a los otros usuarios.	Esta funcionalidad es básica para implementar los servicios de awareness. Se suele tener una ventana adicional indicando la presencia de los usuarios en el sistema. También se suele tener una lista de espacios con las personas que están en ellos. Alguna funcionalidad adicional puede ser la indicación de que actividad están desarrollando y con que elementos. Esta información puede ser el contacto con los otros usuarios para iniciar sesiones de colaboración.
Encuentros. Las personas se encuentran con otras mientras se desplazan entre las habitaciones y pueden iniciar conversaciones ocasionales.	Las listas de usuarios mencionadas en el punto anterior pueden servir para enviar mensajes a otros usuarios. Estos pueden estar en la misma sala o no. Esto permite comunicarse con personas que no están en la misma habitación. También algunos sistemas tienen canales de comunicación automáticos cuando un usuario ingresa a un room particular. Es decir que el usuario se conecta a una sesión de colaboración simplemente entrando en el espacio.
Cantidad de habitantes.	Como en la realidad, los espacios pueden estar vacíos o pueden tener un número grande de usuarios en el. En los sistemas colaborativos la cantidad de usuarios presentes sigue otras reglas. En algunos casos puede estar limitado a un grupo pequeño debido

	a restricciones tecnológicas (En general en aplicaciones colaborativas sincrónicas, como editores colaborativos, la cantidad de usuarios presentes en un ambiente es limitada a cuatro o cinco usuarios ya que se requiere mantener actualizada las interfaces de los usuarios presentes de las acciones que realiza cada usuario. También es importante tener en cuenta cuál es el protocolo de colaboración; por ejemplo los usuarios pueden estar on-line en la misma sesión pero la cantidad de usuarios dependerá si también la posibilidad que tengan los usuarios de participar simultáneamente o no. Si todos pueden escribir al mimos tiempo, la cantidad de usuarios debería ser menor que si escriben por turnos.
Reuniones en tiempo real	El los sistemas colaborativos pueden tenerse reuniones (sesiones) en tiempo real. Se pueden establecer canales de comunicación cuando los usuarios se encuentran dentro de una habitación. Pero como mencionamos en la sección anterior, las reuniones en tiempo real, requieren funcionalidad más compleja para soportar el acoplamiento y el awareness.
Colaboración asincrónica	En los sistemas colaborativos las colaboraciones asincrónicas pueden darse como en la realidad, dejando (o modificando) objetos colaborativos en los contextos compartidos o los usuarios pueden compartir sus producciones enviándoselas usando herramientas convencionales de comunicación como el mail o mensajeros instantáneos. Uno de los ejemplos más comunes de colaboración asincrónica se produce cuando los usuarios colaboran en herramientas como las wikis.

En esta sección vimos la utilidad de modelar aplicaciones colaborativas pensándola en relación a espacios virtuales. El enfoque espacial nos permitirá pensar que lugares "espacios" donde los usuarios hagan determinadas tareas o tengan modalidades diferentes de colaboración. Se realizaron algunas comparaciones con espacios reales y se puntualizaron otras dificultades que los ambientes virtuales introducen. Finalmente vimos como deberían soportarse los espacios virtuales dentro de los sistemas colaborativos.

En este trabajo ponemos mucho hincapié en la utilización de espacios en el diseño de sistemas colaborativos por varias razones. Primero porque el usuario entiende fácilmente la distribución espacial de las cosas. Por otro en los sistemas puede asociarse un espació a una ventana o una porción de la interfaz del usuario. Finalmente porque permite contextualizar las actividades que los usuarios hacen dentro del espacio. La forma de contextualizar es a partir de las herramientas colaborativas que se encontrarán en cada espacio. En el método que se propone en este trabajo contempla la utilización de espacios para modelar las actividades que van a realizar los usuarios.

8. Estado del arte

8.1. Un poco de historia

Los conceptos de sistemas groupware mencionados fueron surgiendo paulatinamente en diferentes aplicaciones prototípicas que marcaron hitos que merecen ser rescatados. Mencionaremos los resultados iniciales que abrieron importantes líneas de investigación que llegan a nuestros días [27]. En 1968 Douglas Engelbart, un adelantado a su tiempo, sorprendió a su audiencia en una sesión especial de una conferencia en San Francisco con una demostración de un sistema llamado NLS (oNLine System, que después fue comercializado como Augmented) [31], que consistía en un prototipo de un sistema de video conferencia, un sistema para compartir la pantalla y un telepointer. La demostración fue una comunicación on-line entre Engelbart en el podio de la conferencia y Bill Paxton desde el laboratorio SRI en Menlo Park. Por otro lado, Engelbart, fue pionero en otras innovaciones, como el uso del Mouse, el editor de dos dimensiones, Windows e Hipermedia. Algunas innovaciones, como Windows, el uso de Mouse y recientemente hipermedia, son ampliamente usadas por millones de usuarios en todo el mundo. En el caso particular de groupware ha tomado más tiempo y seguramente estamos entrando en la etapa en la que se asimilará e incorporará como ambientes habituales de uso por los usuarios de computadora.

En la misma época, en el Instituto de Tecnología de New Jersey, Murria Turoff desarrolló el primer sistema de conferencia basado en computadoras (computer conferencing), llamado *EMISARI* (Emergency Management Information System and Reference Index) que luego en 1976, con algunas modificaciones, se llamó EIES (Electronic Information Exchange System). Por varios años Turoff y Hiltz estudiaron el potencial de los sistemas

de conferencias basados en computadoras, cómo se usan estos sistemas habitualmente y realizaron comparaciones con otros medios de comunicación.

A principios de los ochenta, con la aparición de las computadoras personales, el mundo de las computadoras estaba revolucionado con la interfaces "user friendly" introducidas por Apple Macintosh. Mientras que las computadoras de tiempos compartidos (timeshared computers) – inventadas para darle al usuario la sensación de que eran los únicos en el sistema – mantenían algún servicio elemental de comunicación (básicamente chats), la era de las computadoras personales parecía estar mas lejos que nunca del uso de computadoras en trabajo colaborativo. Recién en 1984, las ideas de Engelbart fueron atendidas en un workshop en el MIT, organizado por Irene Greif y Paul Cashman. Grief estuvo trabajando en el MIT en los sistemas RTCAL y MBlink inspirados en el NLS y Cashman trabajó en Digital Equipment Corporation en el sistema XCP, una herramienta para soportar procesos administrativos que luego veremos como ejemplo de sistema de flujo de trabajo (Workflow Management System). Juntos, Grief y Cashman acuñaron el termino Computer Support Cooperative Work (CSCW) para unificar una nueva área de investigación. A partir de entonces, se sucedieron las principales conferencias CSCW (Computer Support Cooperative Work) sobre este nuevo campo de desarrollo, cada dos años, en Estados Unidos, desde CSCW86 hasta por los menos CSCW08 y en Europa, desde ECSCW89 hasta ECSCW09. Junto a estas conferencias hay muchos otros eventos, tales como workshops y conferencias que producen también mucho material sobre este tema.

A pesar de la gran cantidad de artículos escritos sobre el área y los distintos prototipos de investigación desarrollados, hay pocas herramientas groupware afianzadas en el mundo de las computadoras. Hay muchos aspectos que aún están abiertos e importantes equipos en el mundo están investigando cómo desarrollar ambientes y aplicaciones realmente usables en los distintos dominios.

8.2. Ejemplos de aplicaciones

Tanto en la academia como la industria se han desarrollado aplicaciones groupware que soportan alguna situación colaborativa descripta en las secciones anteriores. Los escenarios colaborativos se definen en función de las tareas que realiza el grupo, la duración de dichas tareas y la organización del grupo desde el punto de vista social y cultural. En este último punto es importante considerar si el grupo maneja un lenguaje común, por ejemplo son colegas de alguna disciplina, o el grupo está abierto a que cualquier tipo de usuario se conecte e intente colaborar en donde se puede dificultar la comunicación. Imaginemos por otro lado un escenario en donde los protagonistas de la colaboración son de distinta nacionalidad.

En la tabla que presentamos a continuación hay una lista de categorías de aplicaciones groupware y algunos nombres equivalentes con los que se los puede conocer en el mercado. El orden de esta lista es similar al orden en que las herramientas fueron dadas a conocer.

Categorías	**Nombres Equivalentes**
computer conferencing systems	bulletin board systems, newsgroup
chat systems	
workflow management systems	office procedure systems, coordination systems
electronic meeting systems	Group (Decision) Support Systems (G(D)SS), electronic meeting rooms
application sharing systems	screen / window sharing systems, desktop / data conferencing systems
shared whiteboards	shared drawing systems

co-authoring systems	collaborative / joint / shared editing systems
multi-user hypermedia systems	
collaborative virtual environments	multiplayer games, virtual worlds
Group scheduling systems	group calendaring systems
Audio conferencing systems	
videoconferencing systems	multimedia conferencing systems
collaborative software engineering systems	

Tabla 1: Sistemas Colaborativos y sinónimos

Describiremos las aplicaciones, enfocando la funcionalidad que se les brindará a los usuarios y comentaremos en cada caso referencias a sistemas representativos.

8.2.1. Computer Conferencing System (sistemas de conferencias, también conocidos como sistemas de noticias)

Estos sistemas son una variante de los sistemas de correo electrónico. Mientras que los sistemas de los correos electrónicos facilitan la comunicación enviando mensajes vía computadoras a uno o más usuarios, los Computer Conferencing System permiten a los usuarios enviar mensajes a un "lugar" en el ciberespacio destinado para un propósito particular. Similar a un pizarrón de novedades, donde se permite abrochar notas para ser leídas luego. Los mensajes enviados a estos sistemas pueden ser recuperados por los interesados con posterioridad. En algunos ambientes de educación a distancia se utiliza este tipo de sistemas para notificaciones específicas dentro de los cursos o en el ambiente en general. Uno de los primeros Computer Conferencing System es el *EMISARI*, desarrollado por Murria Turoff en el New Jersey Institute of Technology, y que luego en 1976 fue conocido como *EIES*.

Uno de los más populares y probablemente mas grande Computer Conferencing System es el *Usenet* (ver Ilustración 1), el cual usa la misma infraestructura de red que se usa en Internet y provee cientos de lugares para discutir distintos temas, conocido como *newsgroup*. Cada uno de estos newsgroups es usado por miles de usuarios y el tráfico de mensajes puede ir desde algunos mensajes a miles de mensajes por día.

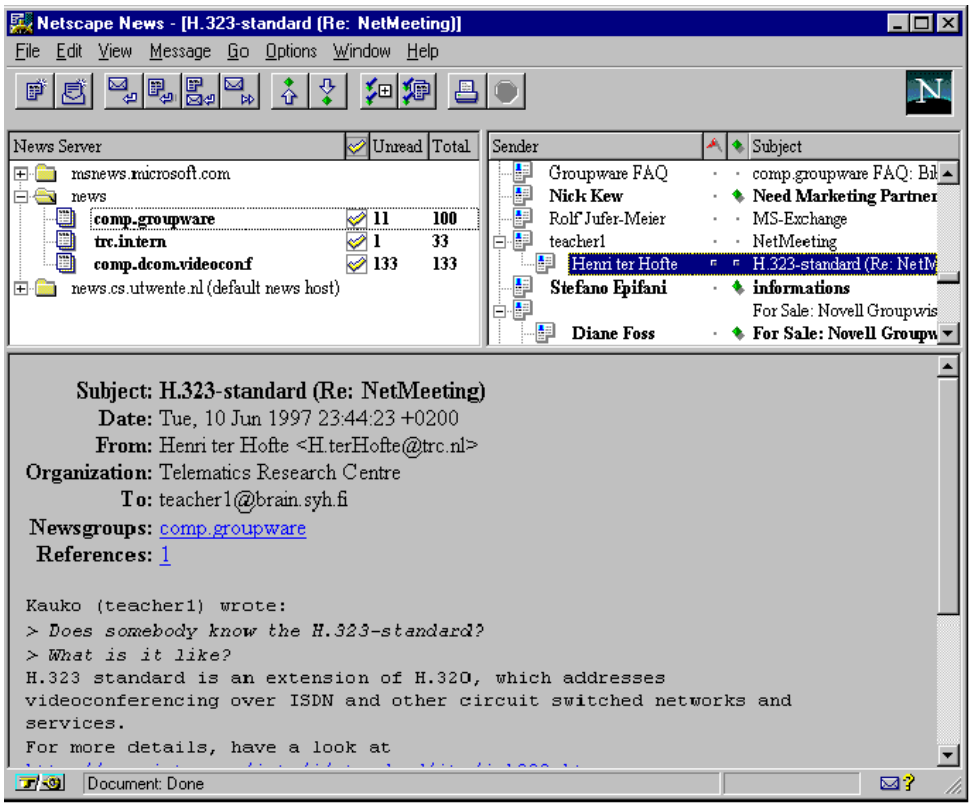


Ilustración 1: Sistema de Noticias

Además de mostrar los mensajes en un orden temporal, algunos sistemas permiten expandir los mensajes explotando relaciones adicionales entre los mensajes. Por ejemplo soportan la relación es respuesta de, el cual permite presentar los mensajes ordenados donde cada mensaje es hijo del mensaje al cual responde. El sistema *gIBIS* soporta cuatro tipos de mensajes – issue, position argument y other – y también las relaciones entre los mensajes como – *questions*, *issuggested-by*, *supports*, *objects-to*, *generalises*, *specialises* y r*eplaces*.

Las primeras versiones de Conferencing System solo permitían enviar mensajes de texto pero las versiones mas recientes, como *Teamtalk*, permiten enviar mensajes de diferente tipo como documentos de procesadores de texto, imágenes, planillas de cálculo, etc. Lotus Notes, uno de los productos comerciales mas exitoso bajo el título de sistemas "Groupware", basa su funcionalidad en funciones de los Computer Conferencing System, soportando documentos de diferentes tipos y provee la posibilidad de programar diversas aplicaciones groupware dedicadas.

8.2.2. Chat Systems (sistemas de chats)

Al igual que los sistemas de E-mail y los Computer Conferencing System, los sistemas de chat proveen comunicaciones entre usuarios, a través de redes de computadoras. La diferencia con los sistemas mencionados es que cada carácter (en algunos sistemas es una frase) tipeado es inmediatamente observado en las pantallas de los otros usuarios, lo que facilita respuestas rápidas en las discusiones. Los primeros sistemas de chat, como el *talk* de *UNIX*, arrancaron como una funcionalidad complementaria de los sistemas operativos de tiempo compartido (timeshared operating systems), que proveían comunicación entre dos usuarios de la red. Luego sistemas como Internet Relay Chat (IRC) permitían comunicación entre varios usuarios conectados a Internet (usualmente usando computadoras personales).

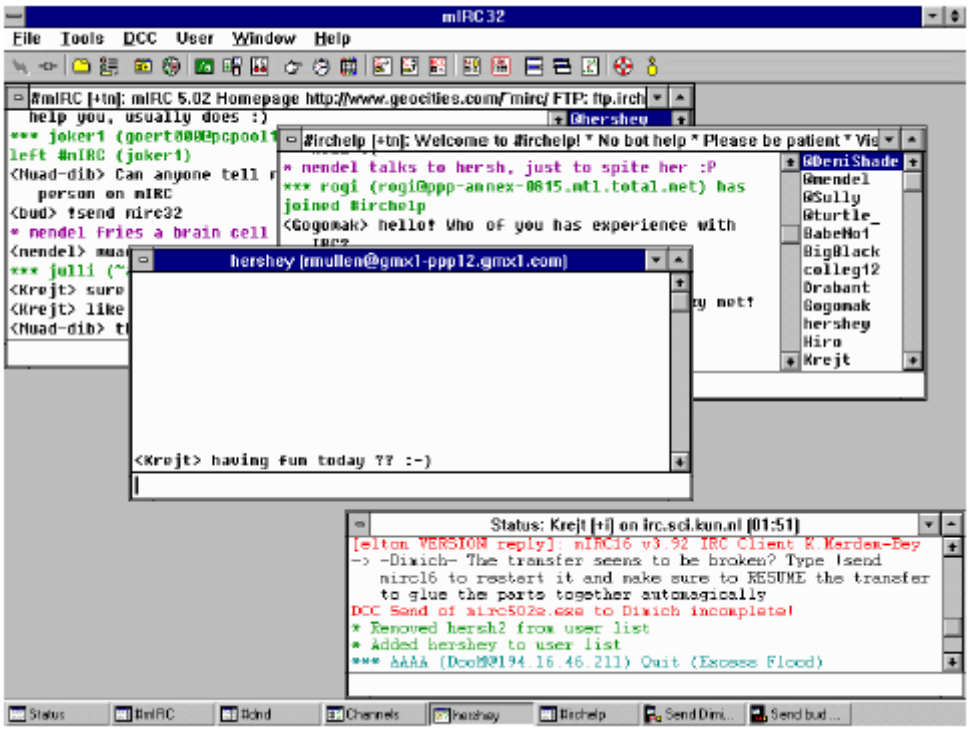


Ilustración 2: Sistema de Chat

8.2.3. Workflow Management Systems (sistemas de manejo de flujo de trabajo)

Los sistemas de workflow se utilizan a un grupo de usuarios a llegar a un objetivo. Estos sistemas coordinan las acciones de los usuarios, alentando o forzando a realizar las acciones en los momentos correctos y por los usuarios adecuados, basados en el modelo de tarea cooperativa que está definido en el sistema workflow. Las primeras investigaciones en esta área como Coordinator, XCP y CHAOS estuvieron basadas en la teoría de la conversación, que considera al lenguaje como una serie de acciones. Estos modelos definen el posible curso de acciones dentro de la conversación entre dos actores.

Otra línea de trabajo está basada en los procedimientos de oficina, los cuales describen las tareas relativamente bien estructuradas. Por ejemplo las tareas necesarias para pedir un

préstamos en términos de roles, sub-tareas y relaciones entre tareas. Ejemplos de estos sistemas son *COSMOS* [71], *AMIGO* [52], *GRACE* [33] y *DOMINO* [54].

Últimamente el interés por los sistemas workflow ha crecido considerablemente y han aparecido una amplia gama de sistemas comerciales, tales como *Staffware*, *InConcert*, *FlowMark* y el desarrollo de estándares por el Workflow Management Committee.

8.2.4. Electronic Meeting System (sistemas de soporte de reuniones)

Usualmente los gerentes y directivos de empresa pasan un amplio porcentaje de su tiempo en reuniones cara a cara. Varios sistemas fueron desarrollados para mejorar la efectividad y eficiencia de estas reuniones (entre ellos aparecen [35], [46], [36]). Uno de los sistemas más simples de este tipo son los sistemas de votación electrónica que permiten un mecanismo para votar rápido y anónimo. Otras herramientas que mejoran el proceso de las reuniones son por ejemplo las de brainstorming con el la Ilustración 3, organización de ideas, evaluación de ideas y recomendaciones.

Usualmente, estos sistemas como *MIS Planning and Decision Laboratory* [34], desarrollado en la Universidad de Arizona — la base del sistema conocido comercialmente como sistema de electronic meeting *GroupSystems* **y** *Colab* [44] desarrollada por Xerox PARC, consiste de una habitación amueblada con pantallas gigantes, conectada a un conjunto de computadoras personales y otros muebles (como sillas y mesas) equipados con computadoras interconectadas. Algunos beneficios que proveen estos sistemas son:

- Tipeo en paralelo y rápida acumulación de ideas. Es más rápido y productivo encontrar ideas interesantes y motivadoras cuando se hace en forma escrita. Mucho mas rápido y productivo cuando los distintos usuarios pueden incorporar las ideas en paralelo (aunque el tipeo sea mas lento que comentar las ideas oralmente). La

presentación oral requiere esperar el momento para hablar y expresar las ideas en forma escrita permite re-escribirlas o refinarlas varias veces antes de enviarlas.

- Anónimo Opcional: Esto puede aliviar factores de inhibición social. Por ejemplo, el anonimato puede facilitar la aparición de ideas brillantes o dar honestas posiciones acerca de algunas ideas.

- Información basada en computadoras puede incorporarse o extraerse de la reunión: Durante las reuniones los usuarios pueden ver y manipular documentos electrónicos (testos, gráficos, tablas). De la misma manera, el sistema puede generar documentos electrónicos como documentos de minutas y lista de acciones a realizar. Este resultado es de gran importancia ya que en muchas de las habituales reuniones que se tienen en las empresas no se produce un resultado escrito de las reuniones.

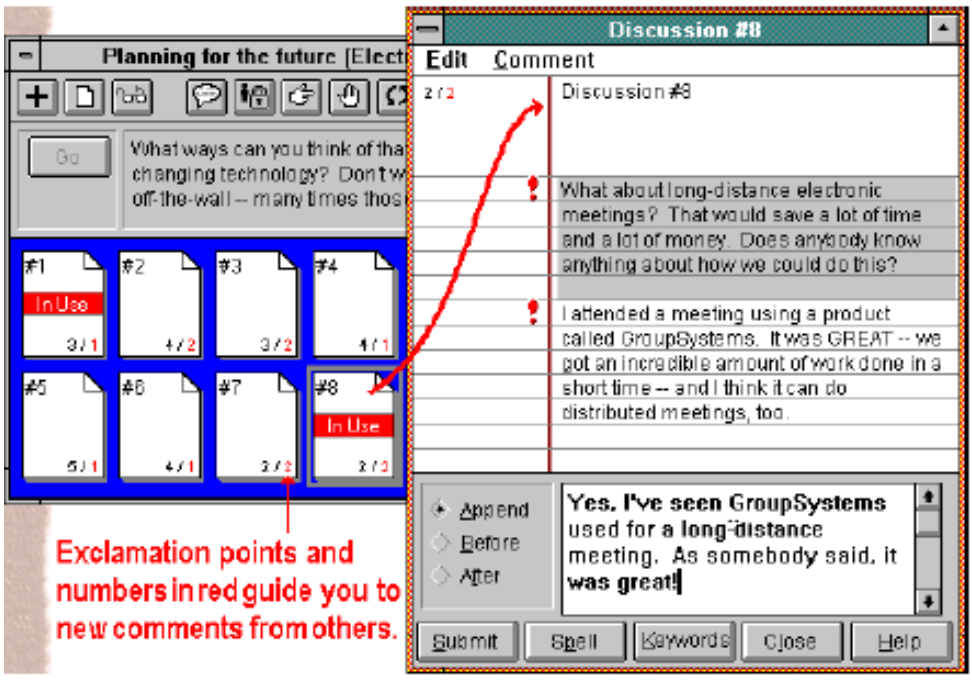


Ilustración 3: Sistema de tormenta de ideas

8.2.5. Application Sharing System (sistema para compartir aplicaciones)

Application Sharing System permite a un grupo de usuarios utilizar simultáneamente (en forma colaborativa) aplicaciones que fueron concebidas como aplicaciones monousuarias. Esto se logra por una funcionalidad genérica externa a dichas aplicaciones que permiten "multicast" la salida (reproducir la salida) de la aplicación y recolectar el input (la entrada) de los usuarios de forma de mantener la imagen de la aplicación monousuario. Usualmente un solo usuarios controla la aplicación por vez y en general estas aplicaciones permiten visualizar todas las acciones de los usuarios, incluyendo los movimientos del mouse o las elecciones que hace en los menues.

En las primeras aplicaciones de este tipo (conocidas como screen sharing system), como NLS/Augment [31], se compartía toda la pantalla y no había espacio privado en las pantallas de los usuarios. Luego con la consolidación de las interfaces gráficas y sistemas orientados a aplicaciones corriendo en ventanas, como Xerox Star en 1981, Microsoft Windows en 1985 y X windows en 1986, apresuraron el desarrollo de aplicaciones de herramientas que permitan compartir la ventana de una aplicación en vez de compartir toda la pantalla. Ejemplo de este tipo de aplicaciones (también conocidas como shared windows o windows shared system) son:

- MBlink4, que estaba basado en compartir la salida de bitmaps de los programas que corrían sobre workstations Xerox Vconf/Dialogo [57], que estaban basados en arquitecturas con terminales gráficas previas al Standard X windows.

- Herramientas basadas en X windows como XTV [68], shX [69] y SharedX disponible comercialmente de Hewlett Packard [72] y el módulo para compartir aplicaciones llamado ShowMe de Sun (ver Ilustración 4);

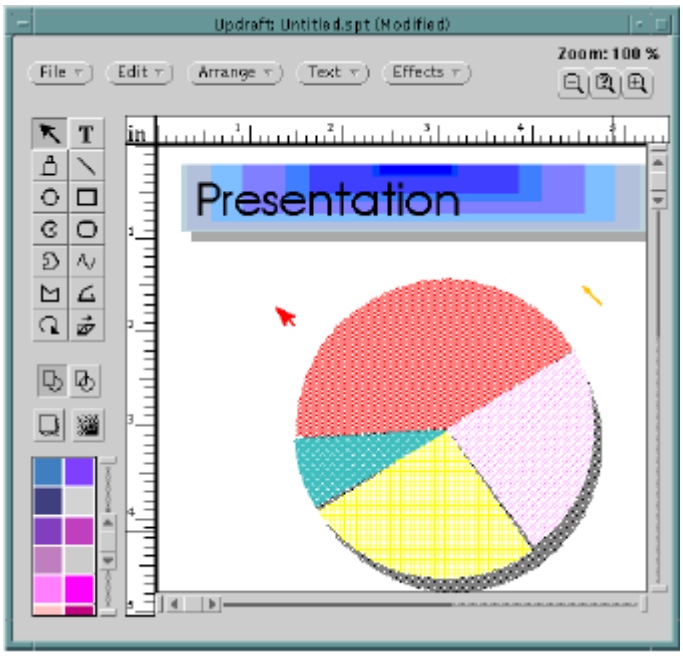


Ilustración 4: Compartir aplicaciones monousuarias

- Microsoft Windows, por otro lado, permite compartir aplicaciones a partir del NetMeeing. En este caso un usuario puede darle el control de su máquina a otro, quien podrá abrir y controlar cualquier programa. Otros sistemas que permiten compartir aplicaciones de Microsoft Windows son el Person to Person de IBM (basado en XTV) [55] y Proshare de Intel.

- Herramientas multiplataforma encontramos a BERKOM Multimedia Collaboration Service (MMC) [29], [42] y a otras dos que están comercialmente disponibles que son Face to Face y Timbuktu pro, que permiten compartir aplicaciones entre usuarios que tiene interfaces gráficas diferentes como Apple Macintosh, MS Windows y X windows.

Estas “application sharing” system últimamente se venden en combinación con sistemas de audio o video conferencia (como es el caso de netmeeting). También suelen incluir algunas aplicaciones como pizarrones compartidos o chats. Estas aplicaciones suelen compartir las aplicaciones transfiriendo el mapa de bits entre los usuarios que participan y

esto hace que sean poco eficientes o requieran un ancho de banda importante para poder operar con ellas. Estas aplicaciones tienen hoy un resurgimiento a partir de los sistemas que comparten las pantallas (llamados remote control system) que permiten hacer soporte de equipos en forma remota.

8.2.6. Shared Whiteboards (pizarrones compartidos)

En muchas reuniones formales o informales, en especial aquellas en las que se discute sobre algún diseño u otros fenómenos complejos, se podrá ver a la gente dibujando sketches (por ejemplo en un pizarrón con en la Ilustración 5), apuntando a ítems o relaciones en particular. Otra gente podrá trabajar con esos dibujos y podrá quizás modificarlo.

Los pizarrones compartidos fueron pensados para soportar estas reuniones, en especial cuando los participantes no se encuentran en la misma habitación. Los objetos que son modificados en el área de trabajo compartida del pizarrón son inmediatamente visibles en todos los usuarios que participan de la sesión. Mencionaremos algunos de los sistemas mas conocidos.

- Basados en Bitmaps encontramos a GroupSketch [41], Wscrawl [42] y disponibles comercialmente a Person-to-Person, ProShare [45], Sun ShowMe y el pizarrón en Netscape Conference. La mayoría de ellos son una variante multiusuario de los programas para editar bitmaps (tipo paint de Windows), que permite a un conjunto de usuarios manipular dibujos. Estos programas suelen tener funciones de importación y captura de imágenes de pantalla o de escáner.

- Entre los vectoriales encontramos a GroupDraw [41], ConversationBoard [62] y los disponibles comercialmente como Aspects y el pizarrón en Microsoft Netmeeting, que son esencialmente una variante multiusuario de los programas monousuarios que permiten manipular dibujos vectoriales que consisten de curvas y figuras definidas matemáticamente en dos dimensiones.

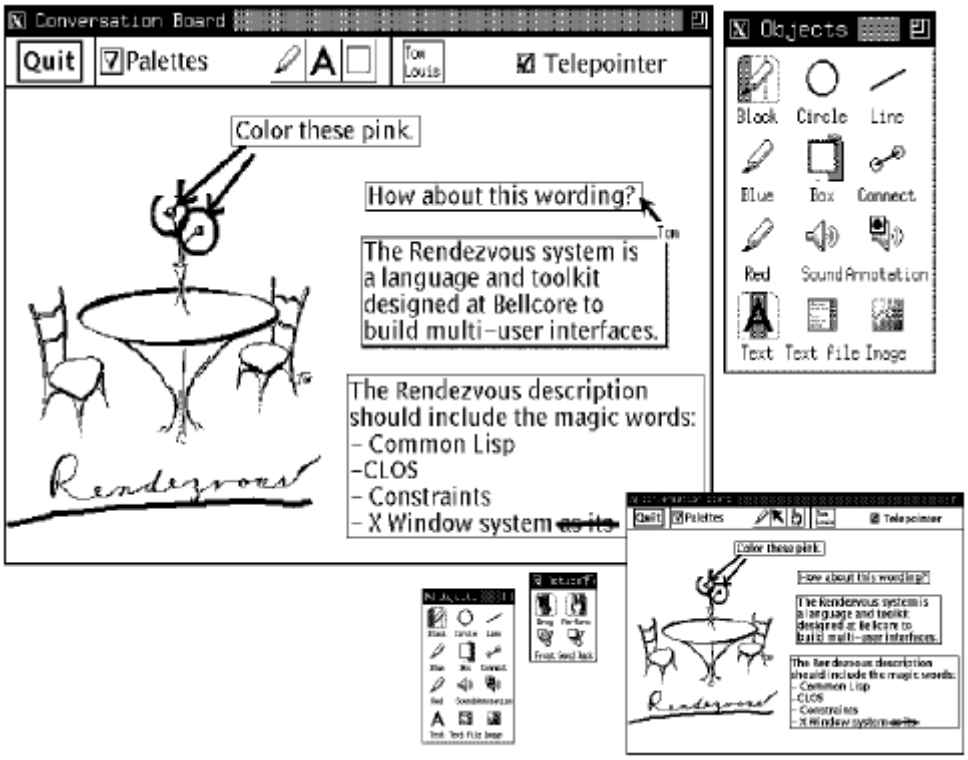


Ilustración 5: Pizarra compartida

Algunas versiones rudimentarias de pizarrones compartidos se pueden obtener usando aplicaciones monousuarios tipo Paint o vectoriales en combinación con los sistemas de application sharing system.

Las versiones mas actualizadas de pizarrones compartidos, permiten que los usuarios dibujar en forma simultánea. En algunos casos también permiten que los usuarios visualicen distintas partes del dibujo que están realizando. (Los usuarios pueden moverse independientemente dentro del dibujo y pueden tener zoom diferente). En estos casos es importante mantener informado al resto de los usuarios, a través del awareness, cual es el área de visualización de cada uno de los usuarios (viewport de cada usuario).

8.2.7. Co-authoring System (sistemas de co-autoría)

Crear documentos es la tarea mas frecuente en el uso de las computadoras. Hasta hace poco tiempo los editores de texto convencionales (como el MS Word) solo asistían al usuario en la producción de versiones lista para imprimir de documentos con muy poco soporte para co-autoría. Aunque muchos trabajos resultan de la cooperación entre uno o mas personas. En algunos campos, como la producción científica, la mayoría de las publicaciones es escrita por dos o más personas. Pese a esto, solo algunos procesadores de texto soportan alguna funcionalidad de control de cambios, comentarios y marcas de revisiones que sirven para la producción de documentos en forma colaborativa. Toda la comunicación y coordinación entre los autores tenía que realizarse sin la colaboración del sistema, por ejemplo pasándose versiones en borrador con la colección de anotaciones o realizando reuniones para discutir en persona algún aspecto del documento.

Versiones rudimentarias de sistemas de co-autoría pueden ser creadas combinando un procesador de texto convencional con un sistema para compartir archivos, un sistema de mensajería o un sistema para compartir aplicaciones.

- Un sistema de mensajería y un sistema para compartir archivos, mejoran la eficiencia sobre sistemas donde se envían los borradores a los autores de los documentos.

- Los sistemas para compartir aplicaciones, combinados con procesadores de texto tradicionales, proveen un ambiente donde un conjunto de usuarios distribuidos en distintos lugares pueden manipular el mismo documento. En este caso no solo el documento es compartido sino que la interfaz del documento también. Los usuarios no pueden hacer scroll, ni tener distintos niveles de zoom ni tipear independientemente. Esto limita la utilidad de este approach a realizar ciertas tareas (como elaborar el outline o revisar alguna sección en particular).

Los sistemas de co-autoría fueron diseñados específicamente para soportar las actividades de un grupo de usuarios para crear un documento en forma conjunta. En los últimos años los procesadores de texto comerciales han agregado cierta funcionalidad de co-autoría tales como combinar las anotaciones de diferentes usuarios, manejo de revisiones y manejo de versiones (las últimas versiones integran su funcionalidad con herramientas de mensajería instantánea) y esta funcionalidad es usada como argumento de venta.

Los sistemas de co-autoría son muy diversos y depende del soporte que tengan para:

- manejo de diferentes fases de autoría, por ejemplo brainstorming, planificación, discusión, revisión, etc.;
- manejo de documentos de solo texto, texto con formato, documentos multimedia;
- edición de documentos en forma simultánea y/o secuencial;
- anotaciones, versiones y revisiones;
- facilidades de comunicación entre usuarios acerca del documento o del proceso de autoría;
- coordinación del proceso de autoría.

Se verá algunos sistemas representativos de co-autoría: Quilt, GROVE, SEPIA, y CoMEDIA.

Quilt fue desarrollado en 1988 en Bellcore [40], [37] y sus diseñadores principalmente focalizaron principalmente que el sistema provea un rico conjunto de mecanismos de anotaciones y la definición de ciertos roles sociales de forma de coordinar el proceso de autoría. La mayor parte del soporte estaba dado en la fase de escribir, editar y revisar.

Quilt trabajaba con una base de datos compartida donde se almacenaba el texto y las anotaciones. Las anotaciones podían ser sugerencias para revisar, y podían ser

comentarios de texto o de voz. Las anotaciones eran vinculadas como links hipertextuales a una parte del documento base en particular o con otras anotaciones. Las anotaciones podían ser de tres tipos: comentarios privados, los que se visualizaban solo al creador del documento, mensajes dirigidos, que eran visualizados por un conjunto de usuarios explícitamente identificados o comentarios públicos. Los usuarios se enterarán de nuevas versiones del documento, sugerencias y anotaciones solo cuando el originador haga "committed" el documento o enviado la sugerencia o la anotación.

Quilt permitía que un grupo de usuarios edite en forma simultánea un documento y ante potenciales conflictos elaboraba versiones de los documentos. Quilt permitía examinar fácilmente las diferencias entre las versiones de los documentos. Para evitar posibles conflictos, Quilt comunicaba un warning cuando otro usuario ya estaba editando el documento base y tenía una opción de locking para evitar esta situación.

La fortaleza de Quilt residía en el amplio soporte para el uso y la definición de roles de usuarios, definidos en una jerarquía de roles que tenían ciertos derechos sobre ciertas acciones sobre los documentos base y las anotaciones. reader < commenter < co-author, donde si un derecho era otorgado para un reader implicaba que ese derecho era otorgado a commenter y a co-autor, pero si cierto derecho es otorgado a commenter, co-author también lo tenía pero no necesariamente era otorgado a reader. Los derechos eran otorgados sobre los siguientes elementos: base document, suggested revision, public comment, directed message, private comment y history. Los derechos soportados por Quilt: create, modify, delete, attach revision, attach comment, attach message, attach private comment y read.

Una colaboración en Quilt es creada por un usuario, tiene un nombre y consiste de un documento, un conjunto de usuarios que son asociados a un rol particular, y el estilo de colaboración esta definido entre los derechos, las unidades y los roles. El creador de la colaboración puede elegir de un conjunto de estilos predefinidos (incluyendo tres provistas por Quilt) o puede crear un estilo de colaboración nuevo.

El Group Outline and Viewing Editor (GROVE) fue desarrollado en el Microelectronics and Computer Technology Corporation (MCC), alrededor de 1988 [32], [38]. Los propósitos originales de este prototipo fueron el de explorar alternativas de implementación de herramientas multiusuarios sincrónicas y el de recolectar observaciones informales de su uso.

Las colaboraciones eran iniciadas por un usuario, el cual especificaba el nombre del archivo de un documento a ser editado. Básicamente cualquier usuario con derechos de acceso a ese documento puede participar en la colaboración simplemente clickeando en el icono del archivo. Los documentos compartidos de GROVE. A diferencia de Quilt, que notificaba los cambios solo cuando los usuarios enviaban o realizaban un commit, GROVE persigue el paradigma de "What You See Is What I See" (WYSIWIS) [44], notificando a los usuarios cada operación individual, por ejemplo (insert, delete) caracteres, (open, close), documentos y (back, forth) barras de scroll. GROVE no emplea mecanismo de locking, y permite operaciones concurrentes. Para mantener la consistencia usa un algoritmo especial de transformaciones operaciones [32].

GROVE permite también relajar el paradigma WYSIWIS. Cada usuario tiene un específico esquema de presentación de ítems que informan al usuario de sus derechos de acceso a los ítems. Asimismo podrá ver solo su propio cursor y verá en diferentes ventanas en diferentes posiciones en la pantalla una estructura de sub-árbol de la estructura general que será visible y accesible a un sub-grupo de participantes. En particular GROVE soporta ventanas privadas (solo accesibles por su creador), ventanas compartidas (accesibles a un conjunto definidos de usuarios) y ventanas públicas (accesible por todos los participantes).

Los usuarios pueden dejar y reconectarse luego a una sesión. En las primeras versiones de GROVE, cuando un usuarios de reconectaba a una sesión, se le mostraba el estado corriente de la sesión. Esto se verá luego como el concepto de "latecomer". En las últimas

versiones se mostraba las modificaciones en el contenido del documento con distintos colores en el texto, ayudando al usuario a reconectarse a la sesión más fácilmente.

GROVE provee control de acceso basado en la estructura del documento, donde se le daba derechos a distintas unidades del documento. Por default, todos los usuarios tienen acceso de escritura y lectura (concurrente) a todo el documento. Sin embargo, para cada parte de los ítems definidos del documento, se puede revocar u otorgar derechos de lectura o escritura a usuarios individuales.

Últimamente algunas herramientas de co-autoría han evolucionado los estilos de Quilt y GROVE. Entre ellas inicialmente encontramos a SEPIA [43] (ver Ilustración 6) y luego VITAL. [67]

SEPIA fue desarrollada en 1992 en la división de Cooperative Hypermedia Systems en el Institute for Integrated Publication and Information Systems (IPSI). Está basado en el modelo cognitivo y su primera versión solo soportaba el trabajo individual. En versiones posteriores, tenía soporte de co-autoría en la modalidad "fuertemente acoplada" y "débilmente acoplada" y posibilidad de cambiar de modo. En 1993 agregaron la funcionalidad para que un usuario pueda trabajar aislado en un documento independiente.

Los diseñadores de SEPIA utilizaron conceptos de Hipertexto para soportar diferentes modos de colaboración brindando soporte para cambiar naturalmente el modo de colaboración cuando se navegaba de un nodo a otro. Los documentos SEPIA son documentos de Hipermedia compuesto por nodos atómicos (que contienen texto, gráficos, imágenes, sonido, etc.), links con títulos y nodos compuestos (estructuras compuestas por nodos atómicos, nodos compuestos relacionados por links). Para garantizar niveles de consistencia, SEPIA utiliza locking automático de nodos y links cuando un usuario selecciona un nodo o un link. Los usuarios pueden realizar un locking explícito para hacer un trabajo independiente en un nodo particular. Cada usuario controla el puntero del

mouse y puede hacerlo visible a otros usuarios[1] con el nombre del usuario atachado para poder identificarlo. Cuando un nodo o un link es seleccionado se indica al resto (a través del soporte de awareness) de los usuarios cambiando el color del objeto seleccionado (amarillo en la pantalla del usuario que seleccionó el objeto y rojo en las pantallas de los otros usuarios). Se puede saber que usuario tiene seleccionado un objeto con una simple consulta al sistema.

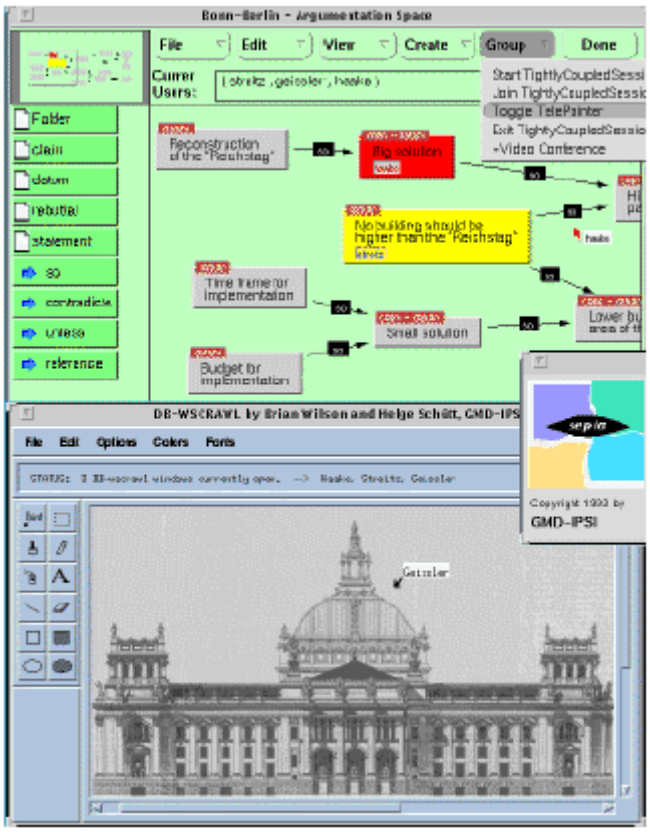


Ilustración 6: Hipermedia colaborativo

Para cada nodo abierto por más de un usuario se abre una ventana separada. Por default los usuarios que abren el mismo nodo entran en el modo "débilmente acoplado". En este modo, las operaciones en los nodos y links son visibles a todos los usuarios y el scroll dentro del documento también independiente. Esto quiere decir que un usuario puede estar visualizando una parte y otro usuario otra. Un grupo de los usuarios pueden dejar el modo

[1] Esta funcionalidad es soportada por el mecanismo de Awareness soportado por SEPIA y se conoce como *telepointer*.

“débilmente acoplado” y pasar al modo “fuertemente acoplado”. Dentro de una sesión fuertemente acoplada, el paradigma WYSIWIS es estrictamente respetado y cuenta para estas sesiones de facilidades de video conferencia. Como un usuario puede estar en una sola sesión “fuertemente acoplada”, entonces no hay posibilidades de superposición de sub-grupos.

SEPIA maneja un elaborado esquema de manejo de versiones, basado en mantener la historia de derivaciones. Permite buscar versiones usando diferentes atributos y luego seleccionar en un browser particular llamado “multi-state object browser”. SEPIA no soporta navegación entre versiones (no soporta el multi-user undo), sin embargo permite combinar y comparar versiones. La comparación puede realizarse siguiendo a un usuario individual o a una sesión “fuertemente acoplada” por un número de usuarios para combinar dos versiones en una nueva versión.

SEPIA no forzaba derechos particulares u obligaciones de acciones particulares sobre los documentos. Es decir, no tiene funcionalidad específica de coordinación de actividades. SEPIA soporta la noción de espacio de trabajo, permitiendo que los usuarios coordinen sus actividades, obligaciones y derechos en el momento. Este espacio de trabajo muestra el desarrollo del proyecto en un momento determinado. Los nodos en el espacio de trabajo corresponden con una versión significativa del documento. Diferentes versiones posteriores pueden derivarse de este documento. Luego estas versiones pueden ser combinadas y obtener una nueva versión. Como el soporte de coordinación de SEPIA está basado en la estructura de Hypermedia, no hay una estructura pre-definida ni un proceso de soporte y la coordinación es definida en el momento por el grupo de usuarios.

Los documentos compartidos de CoMEdiA son un documento de Hipermedia compuesto por capítulos de diferentes medios. CoMEdiA soporta texto, imágenes raster, gráficos 2D, audio y video. CoMEdiA tiene dos tipos de locking (solo se puede usar uno a la vez) que garantizan la consistencia del contenido de los documentos. El chunk lock es una tipo de locking explícito que garantiza la escritura exclusiva a un usuario en una parte del

documento. El position lock por otro lado es un locking automático; el sistema realizará el locking de la porción mínima del documento para que el usuario pueda desarrollar una operación de edición. En CoMEdiA el usuario siempre ve la última versión del documento compartido. El usuario puede tener una posición de scroll independiente en una ventana individual y pueden hacer visible su cursor a un subconjunto de usuarios. También puede filtrar los cursores a ver solo a un sub-conjunto de usuarios (de alguna manera está configurando los elementos de awareness a visualizar). Los usuarios deben seleccionar un cursor del mouse único cuando se juntan en una sesión en CoMEdiA para poder ser identificados. En CoMEdiA los usuarios pueden hacer que su interfaz (su view) sea esclava de lo que ve otro usuario.

Para comunicación directa entre usuarios, CoMEdiA provee de un telepointer compartido, anotaciones, audio conferencia y videoconferencia. Los usuarios pueden tomar el telepointer (y luego dejarlo) para captar la atención de los otros usuarios en una porción particular del documento. CoMEdiA soporta tres tipos de anotaciones: Los comentarios privados y públicos que consisten de porciones de texto o gráficos que son relacionados (hiperlinkeados) al documento. Cuando el usuario prefiere mensajes dirigidos, puede usar la funcionalidad de "off-document communication", que provee una ventana de chat separada para cada par de usuarios.

CoMEdiA provee soporte de coordinación basado en roles predefinidos como chairperson, author, commenter y reader. Cada usuario selecciona un rol cuando ingresa en la sesión. Los roles definen los derechos para realizar las distintas acciones. Un reader puede leer el documento, usar la funcionalidad de realizar anotaciones privadas y comunicarse dentro del grupo (usando off-document communication, audio conferencia y videoconferencia). Un commenter tiene los mismos derechos que el reader, más el derecho de hacer comentarios públicos. Un author tiene los mismos derechos del commenter, más los derechos de editar el documento y puede solicitar información del estado del grupo y otros usuarios. El chairperson (solo puede haber uno por sesión) tiene todos los derechos de los autores mas el derecho de cambiar el rol de los usuarios, editar parámetros particulares de

la sesión, rechazar el ingreso a algunos autores y manipular el documento compartido en general (leer o grabar el documento).

8.2.8. Multi-user Hypermedia Systems (Sistemas de Hipermedia Multiusuarios)

El concepto de Hipermedia, documentos no-secuenciales compuesto por nodos interconectados por links que son recorridos usando el mecanismo de navegación, inspiraron y permitieron la creación de una amplia gama de aplicaciones groupware. Algunos autores sostienen que Hipermedia es potencialmente el medio ideal para soportar el trabajo colaborativo.

Algunos sistemas de co-autoría, como vimos en la sección anterior, están basados en los conceptos de hipermedia, los cuales pueden ser usados como base para dividir las tareas de autoría. El World Wide Web (WWW), por ejemplo, puede ser usado como un sistema simple de co-autoría; cada usuario puede ser responsable de escribir una parte del documento de hipermedia. Los usuarios pueden trabajar simultáneamente cuando trabajan en diferentes nodos. Pueden ver los cambios de los nodos en el momento en que los usuarios los graban. Varios usuarios pueden tener derechos de escritura sobre los nodos, pero generalmente no es simultáneo.

Algunos sistemas de co-autoría, como Quilt y SEPIA usan la funcionalidad de linkear documentos de hipermedia para hacer anotaciones y revisiones del documento principal. Algunos soportan otros tipos de links entre los nodos, por ejemplo, permiten establecer relaciones de planificación o jerárquicas, que permiten diferentes visualizaciones sobre los documentos.

Otros sistemas como NSCA Hypernews [56], ComMentor y otros sistemas de conferencia, usan la infraestructura web; los mensajes son linkeados a una página web.

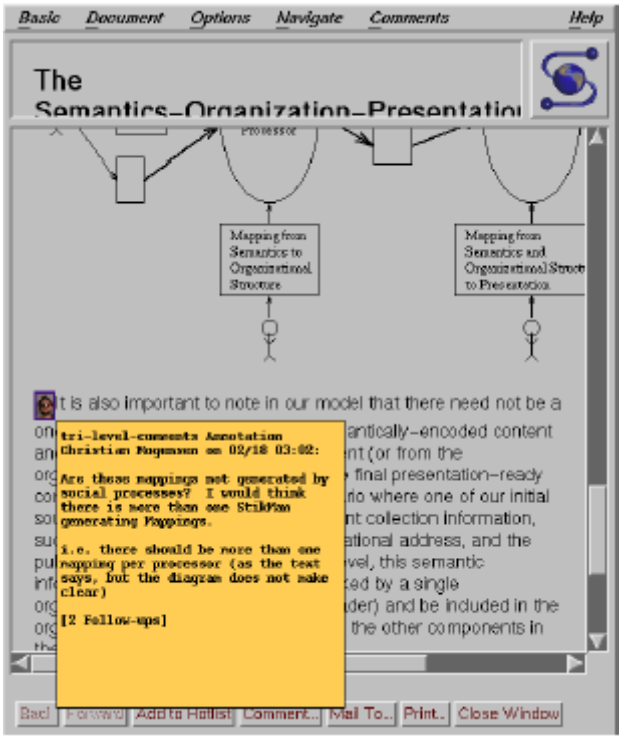


Ilustración 7: Colaboración en la Web

8.2.9. Collaborative Virtual Environments (Ambientes Virtuales Colaborativos)

Comprende las aplicaciones de juegos multiusuario de red o de simulación de batalla distribuida. Los juegos de computadora son un negocio muy importante y merecen pegarle una mirada para ver que podemos aprender de ellos. Es sorprendente la larga historia que tienen estos juegos y han sido una gran inspiración y validación de investigaciones en la ciencia de la computación. Por ejemplo los juegos de ajedrez fueron estudiados y sirvieron de validación por largas investigaciones de inteligencia artificial y procesamiento paralelo. Así también en otras áreas como el paradigma de manipulación directa dentro del área de Human Computer Interaction.

La industria de desarrollo de juegos de computadoras están cambiando del paradigma monousuarios al paradigma multiusuario impulsado por las facilidades de interconexión

en redes locales entre las computadoras personales (tanto en ambientes de oficinas como en negocios de juegos en red) o en una escala global entre computadoras personales a través de Internet. Hay algunos, como el World Chat, son un caso particular de 3D-spaces, donde los usuarios navegan por un espacio (son representados en el sistema con un avatar, que es una imagen que los representa) y pueden interactuar con otros usuarios que están cerca de él con una herramienta de chat textual. En algunos casos como la aplicación del museo de guggenheim los usuarios pueden interactuar entre ellos o con guía (no humano) al cual se le puede preguntar alguna cosa sobre las obras o artistas del museo. Algunos de los ejemplos de juegos fueron casos de pruebas (test cases) de tollkits para el desarrollo de aplicaciones groupware, por ejemplo el Tic-Tac-Toe y CardTable fueron realizados con Rendevouz [53] y Tic-Tac-Toe, Solitaire y Terrominos fueron desarrollados con GroupKit [21].

Además de usar este tipo de aplicaciones para juegos, también son usados para la enseñanza en educación y entrenamiento. Como un ejemplo se menciona a las simulaciones de batallas distribuidas son usadas en el entrenamiento de algunas fuerzas armadas como herramienta de entrenamiento.

8.2.10. Group Scheduling Systems (Sistemas de Agendas de Grupo)

Armar reuniones en un grupo es una de las tareas donde el soporte de aplicaciones colaborativas podría brindar muchos beneficios. En particular, encontrar tiempo libre de varias agendas de distintas personas puede ser realizada mas eficientemente por aplicaciones groupware. Uno de los primeros sistemas para armar estas agendas de grupo fue el RTCAL, como una extensión de la agenda personal PCAL que se usaba en el MIT. Este permite que un número de usuarios puedan alinear las agendas y armar reuniones y que podía ser soportada también por el mecanismo del voto. No tenía función de encontrar tiempos libres. Herramientas mas contemporáneas como el MS Schedule+ (ver Ilustración 8), Lotus Calendar y Novell Groupwise Calendar las cuales tienen esta función de

búsqueda de tiempo libre. Sin embargo, una reunión en estos sistemas no requiere que los participantes estén disponibles durante todo el proceso de armado de la reunión, involucra el intercambio de una serie de mensajes especiales como invitación, aceptación o rechazo.

Pero no todo es color de rosa en este tipo de aplicaciones [30]. Los usuarios deben mantener actualizadas sus agendas personales para que el sistema pueda trabajar con datos fehacientes. En estos casos, la aplicación fallaba porque los usuarios tenían que realizar un trabajo adicional y poco se beneficiaban del sistema. En algunas casos donde las organizaciones tiene una estructura jerárquica, los estratos superiores de la jerarquía se beneficiaban, ya que usualmente tienen personal que les mantiene actualizadas sus agendas.

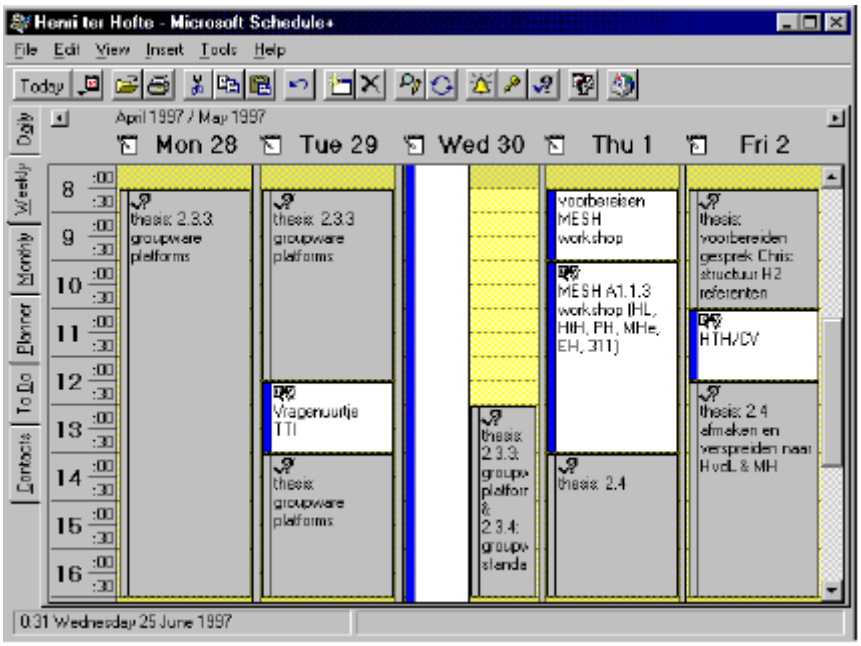


Ilustración 8: Agendas compartidas

8.2.11. Audio Conferencing System (Sistemas de Audio Conferencia)

En el diseño de muchas aplicaciones colaborativas, como las de co-autoría o pizarrones compartidos, se asume que los usuarios podrán comunicarse vía algún sistema externo de

vos (puede ser en algunos casos por un sistema de telefonía convencional). De acuerdo a los avances tecnológicos el soporte de audio streaming sobre redes de computadoras como Internet. Otros sistemas groupware como Netscape Conference, y Microsoft Netmeeting proveen audio conferencia (de vos sobre IP) como un servicio auxiliar integrado con el pizarrón compartido y el sistema de compartir aplicaciones. Este aproach permite armar espacios compartidos y el sistema de audio conferencia, marcando cual de los usuarios está hablando con una indicación en la pantalla, sincroniza el audio y coordinando las acciones en el espacio de trabajo (workspace).

8.2.12. Videoconferencing System (Sistemas de Video Conferencia)

Desde la introducción de la video telefonía en las películas de ciencia ficción y la introducción de Picturephone de AT&T en 1964, los diseñadores han explorado el uso de video para soportar actividades colaborativas a distancia. Estas investigaciones han desarrollado varios prototipos de video conferencia enfocando diferentes aspectos.

- Sistemas como CRUISER [59] and Montage [63] permite explorar si hay alguna persona en la oficina está disponible para conversar

- Sistemas como VideoWindows [70], el cual es usado como una gran pantalla permanente de conexión entre dos lugares geográficamente separados donde la gente generalmente realiza comunicaciones informales. En algunas instituciones se usa para comunicar (integrar) a la gente en dos cafeterías de oficinas distantes.

- Algunos sistemas proveen integración de componentes de video dentro de espacios compartidos (workspaces) en sistemas de conferencia que usan multimedia como ClearBoard. Este sistema permite, desde hace largo tiempo, una integración de dos personas en una sesión de dibujo compartido, como si ellos estuvieran dibujando a

ambos lados de un vidrio transparente, donde pueden verse y ver que acciones están realizando cada uno. (ver Ilustración 9)

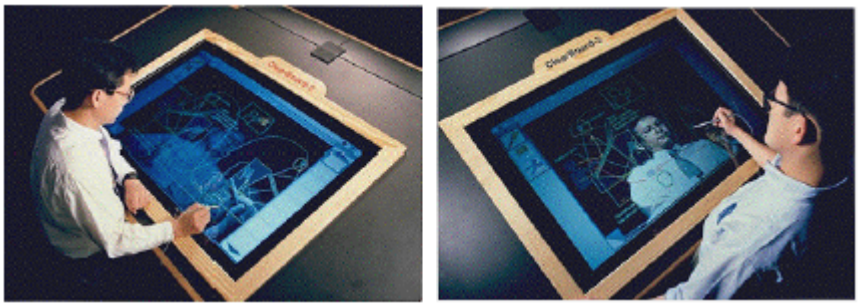


Ilustración 9: Edición colaborativa

- Algunos prototipos avanzados toman esta misma idea de integración entre video y espacio de trabajo, pero lo llevan un paso mas adelante proveyendo videoconferencia en 3D. Un ejemplo es el sistema desarrollado por la Advanced Telecommunication Research (ATR) Laboratory en Japón [47] que provee una ilustración de un espacio 3D que pueden ser vistos por dispositivos especiales. Últimamente los ATR intenta poner la representación 3D de otros participantes en el espacio 3D también, dando de esta forma una completa videoconferencia virtual en el espacio.

Facilidades de la tecnología de redes públicas como Integrated Services Digital Network (ISDN), el uso de tecnologías Asynchronous Transfer Mode (ATM) y el desarrollo de tecnologías de compresión de imágenes y sonido permiten que surjan productos comerciales de videoconferencia. Podemos distinguir tres sistemas de videoconferencia de acuerdo a su apariencia física.

- Sistemas de videoconferencia orientados a ambientes: Consisten de habitaciones dedicadas, preparadas con grandes pantallas, cámaras y micrófonos. Una reunión en estos ambientes, típicamente consiste de dos habitaciones con un grupo de personas en cada habitación que pueden verse unos a otros. Este sistema

finalmente fue utilizado por grandes organizaciones que podían afrontar el costo de comunicaciones costosas.

- Sistemas de videoconferencia portátiles: Son similares a los sistemas de videoconferencia orientados a los ambientes, con la diferencia que los equipos son de fácil transportación. Hay equipos preparados con el hardware (pantalla, cámara y micrófono) y el software necesarios en un rack que puede conectarse a un ISDN.

- Sistemas de videoconferencia desktop: Permiten realizar sesiones de videoconferencia entre dos individuos usando las computadoras personales. Estas videoconferencias no requieren ambientes con grandes pantallas ni de equipamiento dedicado para videoconferencia. Sirven para interacciones cortas en directo entre dos (en algunos casos pueden ser mas de dos participantes de la videoconferencia). Algunos ejemplos comerciales de videoconferencia desktop son CU-SeeMe de White Pine Software, la serie Live de productos de videoconferencia de PictureTel, Proshare de Intel, Showme de Sun, InPerson de Silicon Graphics y NetMeeting de Microsoft. Usualmente los sistemas de videoconferencia desktop se venden como "multimedia conferencing system" e incluye un pisaron compartido y/o un sistema para compartir aplicaciones. Estos sistemas de videoconferencia pueden usarse sobre una amplia gama de redes como ISDN, LAN (Local Area Networks) incluso Internet, pero requieren un mínimo ancho de banda garantizado para que la comunicación no se vea degradada.

8.2.13. Collaborative Software Engineering System (Sistemas de Ingeniería de Software colaborativo)

El desarrollo de software (especialmente el desarrollo de aplicaciones no triviales) es un proceso colaborativo donde interviene un grupo de personas. La mayoría de los herramientas de Ingeniería de Software, sin embargo, son monousuarios (algunas de ellas brindan algunas herramientas complementarias de comunicación entre los

desarrolladores). Hay varias iniciativas que tienen el objetivo de proveer soporte colaborativo a la ingeniería de software cuyo resultado se vio reflejado en los siguientes prototipos:

- ICICLE [49], el cual permite inspeccionar código en grupos.
- GroupCRC [50], el cual soporta el diseño Orientado a Objetos de tarjetas CRC, en el que los participantes asumen el rol de objetos participando en un escenario con el objetivo de entender grupalmente el diseño y descubrir defectos o responsabilidades faltantes o colaboraciones.
- Serendipity/SPE [48] y SPADE-I [51], los que apuntan a una integración del proceso de modelización (process-modelling) al de ingeniería de software por un lado y el soporte de comunicaciones en tiempo real provistas por varias aplicaciones groupware. Al igual que los sistemas de manejo de workflows, los ambientes de ingeniería de software centrada en procesos (PSEEs) contienen un modelo de proceso cooperativo el cual es usado para guiar y chekear las acciones de los usuarios.

En esta sección se comentó el origen de lo que hoy llamamos la tecnología groupware y se recorrió la historia de aplicaciones que fueron hitos importantes en la evolución de los sistemas colaborativos. Se notó que los ejemplos presentados no tenían una visión integral de la problemática y que enfocaban un aspecto particular. Muchos estaban focalizados en la comunicación otros en la coordinación de tareas y algunos pocos hacían hincapié en la edición colaborativa. Los ejemplos presentados son considerados hoy por hoy como herramientas colaborativas que requieren ser integradas para brindar un conjunto de servicios colaborativos.

En este trabajo se presenta un método (ver capítulo Método de Especificación) en donde se pretende diseñar ambientes que contemplen la utilización de distintas herramientas para soportar distintas actividades colaborativas en un mismo entorno colaborativo.

9. Lenguaje de modelado de sistemas colaborativos (CSSL - Collaborative Software System Language)

Un modelo de un sistema colaborativo es una descripción del sistema escrito en un lenguaje bien formado, generalmente gráfico, el cual esta preparado para ser interpretado automáticamente. La técnica más común para la especificación de la sintaxis de un lenguaje gráfico se llama *metamodelado*. Un metamodelo define que elementos pueden existir en un modelo. Por ejemplo el metamodelo de UML define que podemos usar los conceptos de "*Class*", "*State*", "*Package*", etc. en un modelo UML.

Los Metamodelos son también modelos gráficos, por ello un metamodelo en si mismo tiene que ser escrito en un lenguaje bien definido. Este lenguaje se lo denomina metalenguaje. En teoría se puede encontrar un metamodelo de un metamodelo, que se llamaría meta-metamodelo y así en forma infinita. Pero la OMG usa cuatro niveles M0, M1, M2, M3 que están representados en la Ilustración 10.

9.1. ¿Por qué es necesario un metamodelo?

- Permite definir la sintaxis abstracta y semántica de los elementos del dominio, facilitando el entendimiento y comunicación de los modelos.
- Permite el intercambio de modelos entre las herramientas de modelado.
- Permite la representación de elementos del dominio específicos.

9.2. Niveles de abstracción de la OMG

OMG (Object Management Group) define una arquitectura basada en cuatro niveles de abstracción que van a permitir distinguir entre los distintos niveles conceptuales que intervienen en el modelado de un sistema. Esos cuatro niveles son:

1. Meta metamodelo
2. Metamodelo
3. Modelo del usuario
4. Instancias en tiempo de ejecución

La responsabilidad primaria de la capa de meta-metamodelo es definir el lenguaje para especificar un metamodelo. Esta capa es conocida como M3, MOF (Meta Object Facility) [14] es un ejemplo de un meta metamodelo. Por lo general este es más compacto que un metamodelo, y a menudo define varios metamodelos.

Un metamodelo es una instancia de un meta metamodelo y significa que cada elemento del metamodelo es una instancia de un elemento del meta metamodelo. La responsabilidad primaria de la capa del metamodelo es definir un lenguaje para especificar modelos. Esta capa es conocida como M2; UML (Unified Modelling Language) y OCL [15] (Object Constraint Language) son ejemplos de metamodelos. En general estos son más detallados que los meta metamodelos que los describen, sobre todo cuando ellos definen semántica dinámica.

Un modelo es una instancia de un metamodelo. La responsabilidad de esta capa es definir un lenguaje que describa los dominios semánticos, es decir, para permitirles a los usuarios modelar una variedad de dominios diferentes, como procesos, requerimientos, etc. Esta capa es conocida como M1. El modelo del usuario contiene los elementos del modelo y los snapshots de instancias de dichos elementos.

Por último la capa de instancias es conocida como M0, representa las instancias de los elementos del modelo en tiempo de ejecución. Los snapshots que son modelados en M1 son versiones reducidas de las instancias en tiempo de ejecución. Los niveles de abstracción de la OMG la podemos apreciar en la Ilustración 10.

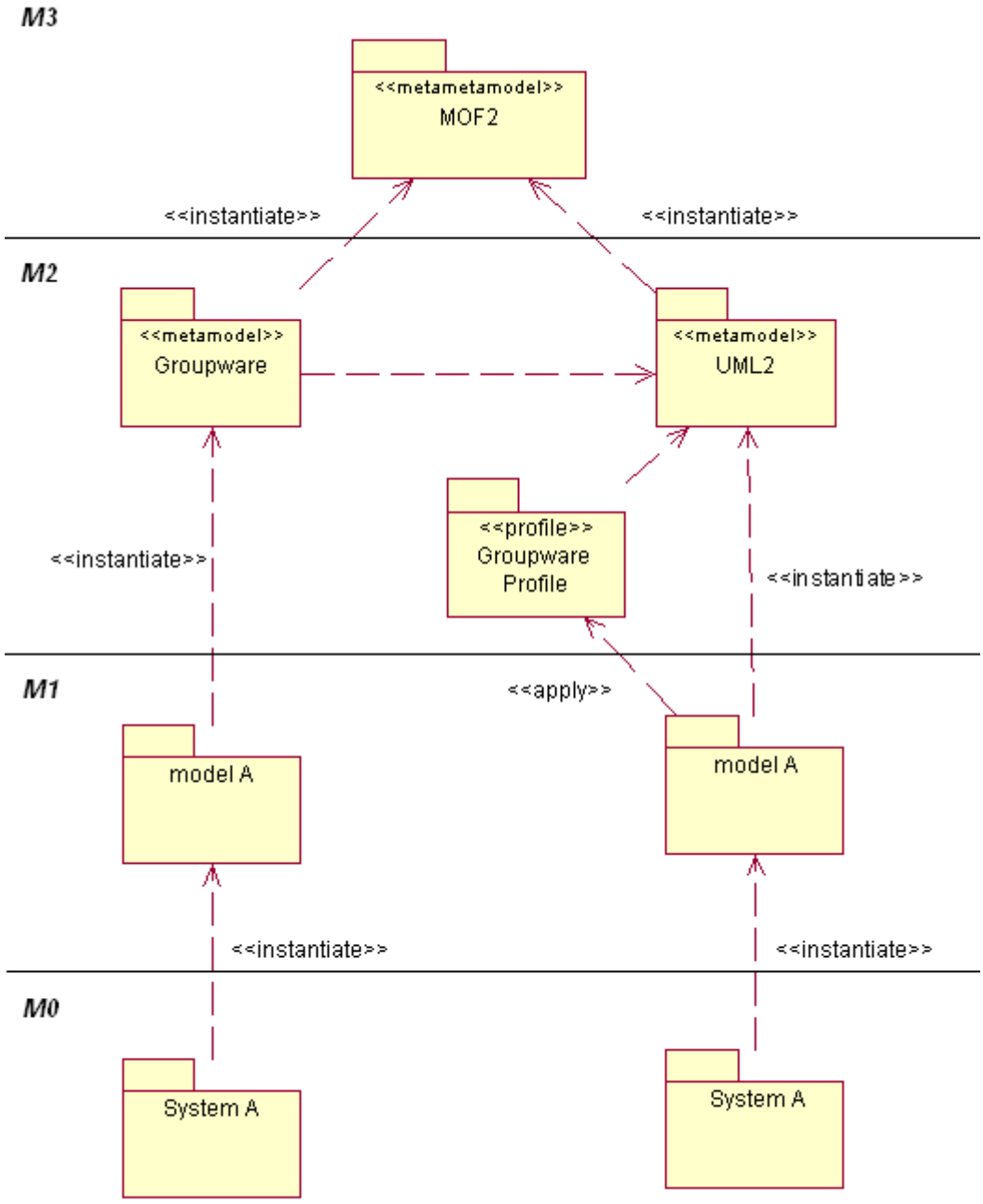


Ilustración 10: Niveles de abstracción de la OMG

UML proporciona un conjunto de mecanismos para extender y adaptar las metaclases de un metamodelo cualquiera (y no sólo el de UML) a las necesidades concretas de una plataforma (por ej. J2EE) o de un dominio de aplicación (por ej. groupware). De este

modo es posible adaptar algunos de sus conceptos, sin necesidad de definir un nuevo lenguaje utilizando el MOF.

9.3. Metamodelo

9.3.1. ESTRUCTURA

El metamodelo de sistemas colaborativos se estructura en 3 paquetes tal y como queda reflejado en la Ilustración 11.

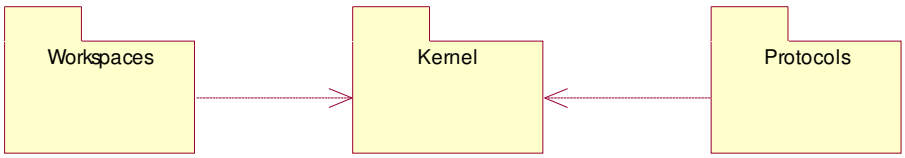


Ilustración 11: Diagrama de paquetes del metamodelo de sistemas colaborativos

Estos paquetes nos van a proporcionar las siguientes capacidades:

- Kernel: Contiene todas las clases y abstracciones que constituyen la base para el resto de los paquetes del metamodelo.
- Workspaces: Contiene las clases necesarias para la creación de modelos de entornos colaborativos.
- Protocols: Contiene las clases necesarias para representar la parte dinámica del sistema.

Los paquetes mencionados anteriormente importan clases del metamodelo UML2.

9.3.1.1. Kernel

El paquete Kernel constituye, tal y como se dijo en el apartado anterior, la base sobre la que se construyen el resto de los paquetes del metamodelo. Sus clases principales son

CollaborationRole y SharedObject. La clase User representa a los usuarios dentro del sistema. En general los elementos colaborativos tendrán relación con el ColaborationRole, ya que un usuario va a tener más de un rol durante el desarrollo de las actividades colaborativas. También aparece la clase CollaboraiveAssociation que se utilizará luego para relacionar los elementos colaborativos y que es subclase de Association de UML. Los elementos de este paquete quedan descriptos en el metamodelo que podemos apreciar en la Ilustración 12 e Ilustración 13.

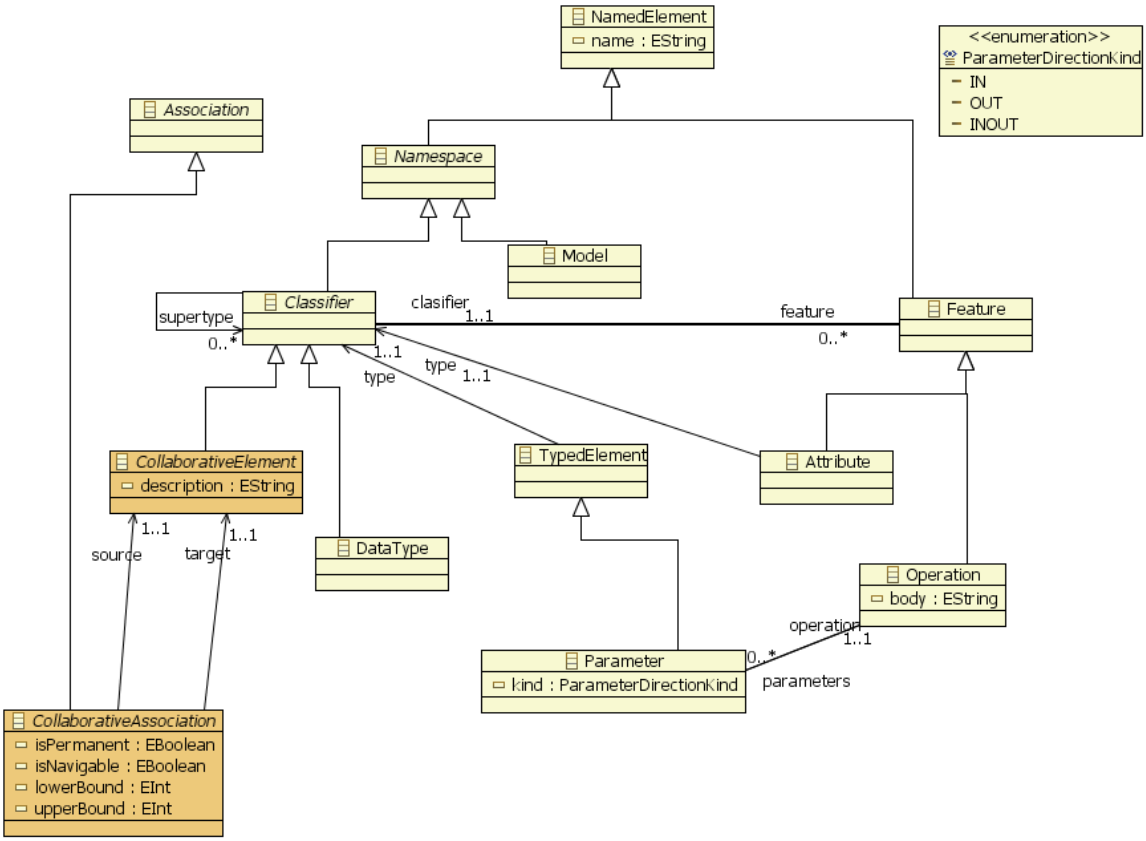


Ilustración 12: Estructura del paquete Kernel

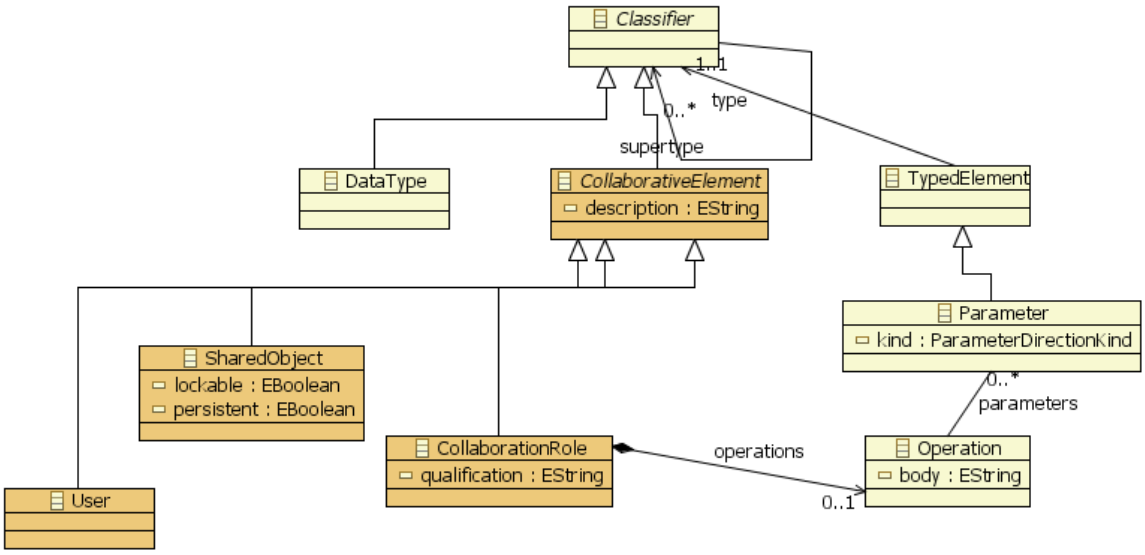


Ilustración 13: Diseño de la jerarquía de los elementos colaborativos – Kernel

Se detallan a continuación las clases principales de la estructura del Kernel

CollaborativeElement		
Super Class:	Classifier	
Descripción:	Clase abstracta que generaliza cualquier elemento que forma parte de un modelo colaborativo – describe un conjunto de instancias que tienen características comunes.	
Propiedades:		
description	**String**	Es una descripción de la clasificación.
Constraints:		
Profile Notation:		

Stereotype	**Superclass**	**Keyword**	**Properties**	**Abstract**	**Icon**

Collaborative Element	*Classifier*		*description*	*SI*	

Collaboration Role		
Super Class:	CollaborativeElement	
Descripción:	Conjunto de habilidades, competencias y responsabilidades de un participante. Las responsabilidades de los roles se especifican como operaciones que se pueden llevar a cabo.	
Propiedades:		
qualification	**String**	Especifica las competencias que debe tener el rol.
/sharedObjects	**SharedObject**	Representa los objetos que están asociados al rol. En este caso serán los objetos compartidos que usa este rol. `Context CollaborationRole def: sharedObjects = self.useRelationship.target-> select (p| p.oclIsKindOf(SharedObject))`
/tools	**Tool**	Representa las herramientas que están asociadas al rol. Es decir las herramientas que puede usar el rol. `Context CollaborationRole def: tools = self.useRelationship.target-> select (p| p.oclIsKindOf(Tool))`
/responsibilities	**Operation**	Especifica las responsabilidades (operaciones permitidas) del rol. `Context CollaborationRole def: responsibilities = self.operations`
Constraints:		
Profile Notation:		

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
Role	*CollaborativeElement*	*Role*	*description*	*No*	

Shared Object		
Super Class:	CollaborativeElement	
Descripción:	Es cualquier cosa que los usuarios puedan crear. Los usuarios también pueden usar los objetos colocados por un tutor en su ambiente o sesión para aprender algo como ser papers, páginas web, etc.	
Propiedades:		
lockable	**Boolean**	Especifica si el objeto compartido es lockable. El valor por defecto es falso.
persistent	**Boolean**	Especifica si el objeto compartido es persistente. El valor por defecto es falso.
Constraints:		
Profile Notation:		

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
SharedObject	*CollaborativeElement*	*shared object*	*lockable, persistent*	*No*	

User	
Super Class:	CollaborativeElement
Descripción:	Para representar quién realiza o participa en una actividad. Se configura en forma persistente.

Propiedades:		
avatar	**Image**	Es la representación de un usuario dentro del sistema.
Constraints:		
Profile Notation:		

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
Role	*CollaborativeElement*	*Role*	*avatar, roles*	*No*	

9.3.1.2. Workspaces

El paquete Workspaces contiene los elementos estructurales de los entornos colaborativos como espacios de trabajo, sesiones y herramientas y las relaciones entre los elementos colaborativos. Su objetivo es definir la estructura del espacio donde se va a efectuar la colaboración. Los elementos de este paquete quedan descritos en el metamodelo que podemos apreciar en la Ilustración 14 e Ilustración 15.

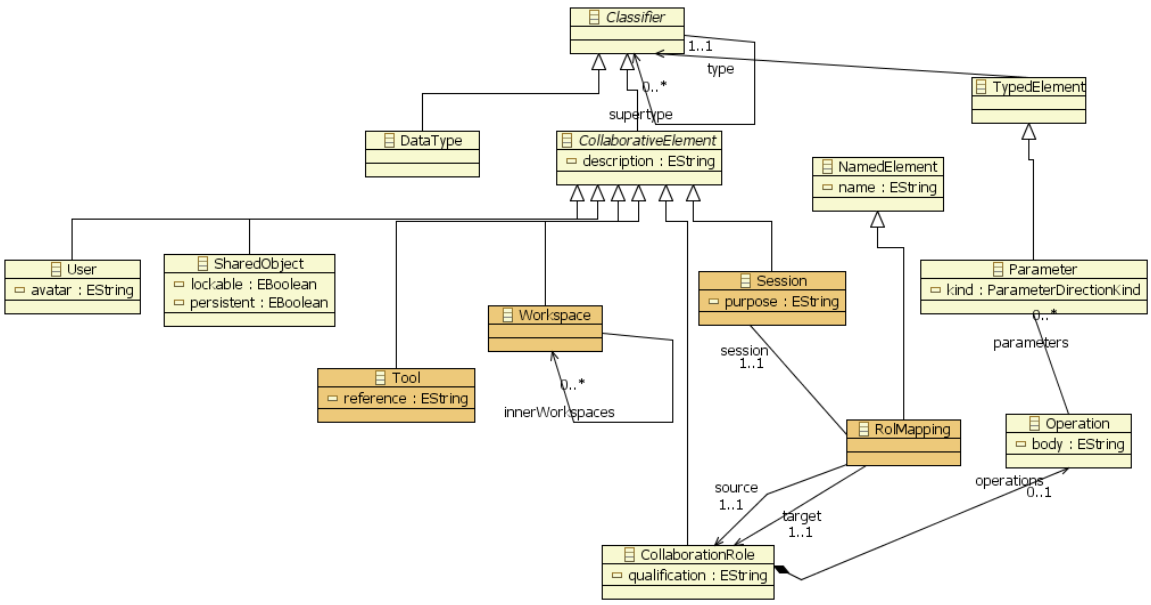


Ilustración 14: Elementos Colaborativos del Workspace

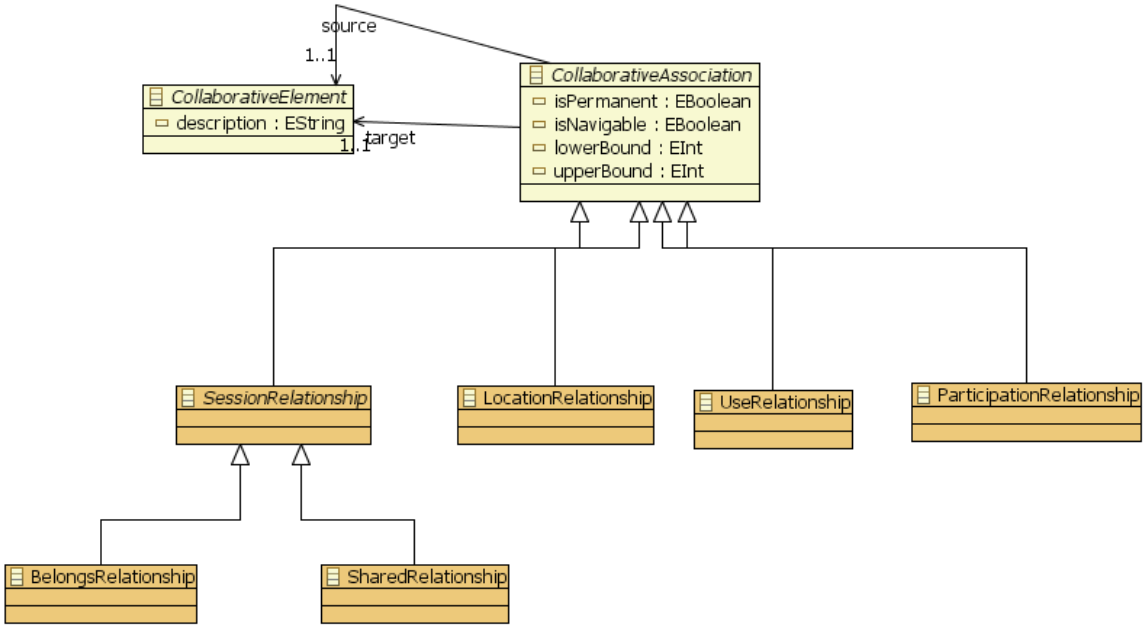


Ilustración 15: Asociaciones entre elementos colaborativos

Se detallan a continuación las clases del paquete Workspace

Session	
Super Class:	CollaborativeElement
Descripción:	Una sesión es un período de intercambio de información soportada por un sistema groupware. La sesión se establece espontáneamente por ejemplo cuando dos usuarios inician una conversación con el chat o también puede fijarse para un periodo de tiempo definido. En general las sesiones mantienen un estado ya que alguna de las partes involucradas necesite guardar información que se podrá usar durante el periodo en que la sesión esté activa. Por ejemplo si queremos saber quienes son los usuarios que están conectados.

Propiedades:		
purpose	**String**	Especifica el propósito de la interacción.
/sharedObjects	**SharedObject**	Representa los objetos compartidos que están asociados a la sesión. `Context Session def: sharedObjects = self.collaborativeAssociation.target-> select (p|p.oclIsKindOf(Tool))`
/tools	**Tool**	Representa las herramientas que están asociadas a la sesión. `Context Session def: tools = self.collaborativeAssociation.target-> select (p|p.oclIsKindOf(Tool))`
/participants	**ColaborationRole**	Representa los participantes que están asociados a la sesión. `Context Session def: participants = self.collaborativeAssociation.target-> select (p| p.oclIsKindOf(ColaborationRole))`
Constraints:		
Profile Notation:		

Stereotype	**Superclass**	**Keyword**	**Properties**	**Abstract**	**Icon**
Session	*CollaborativeElement*			*No*	

Workspace	
Super Class:	CollaborativeElement
Descripción:	El Workspace es el lugar en el que la colaboración se lleva a cabo. Dentro de un workspace hay herramientas colaborativas que son utilizadas para comunicarse y trabajar con los objetos compartidos.

	La participación de los usuarios estará guiada por las actividades que podrán realizar dentro de las sesiones de colaboración y los protocolos que estructuran las interacciones entre los roles en el workspace y habilitan el uso de las herramientas en cada momento. Los espacios de trabajo pueden estar compuestos por sub-espacios que a su vez tendrán herramientas, sesiones y protocolos que permiten componer espacios de colaboración más complejos.

Propiedades:

innerWorkspace	**Workspace**	Representa los workspaces que componen al espacio de trabajo compuesto.
/isAtomic	**Booleam**	`Context Workspace def: isAtomic = self.innerWorkspace -> isEmpty()`

Constraints:

Profile Notation:

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
AtomicWorspace	*Setting*			*No*	

Por otro lado aparece la jerarquía de asociaciones que representan relaciones con cierta semántica entre las clases derivadas de elementos colaborativos. Por ejemplo expresan la ubicación de un elemento dentro de un workspace o la posibilidad de un rol de usar alguna determinada herramienta. Describimos a continuación las clases de asociaciones detalladas en el meta-modelo.

ColaborativeAssociation	
Super Class:	Asociation
Descripción:	Es una metaclase abtracta. Representa diferentes asociaciones entre los elementos colaborativos. Por ejemplo relaciones de ubicación de los elementos colaborativos en algún espacio,

	participación de algún rol en alguna sesión o la posibilidad de uso de alguna herramienta. En el caso de las sesiones podemos encontrar relaciones de dependencia o referencia que sirven para distinguir si una sesión es compartida o no.

Propiedades:

isPermanent	**Boolean**	Indica si la asociación es permanente o transitoria. Por ejemplo sirve para que algunos objetos puedan cambiar de lugar. Estarán asociados temporalmente a un espacio y luego se moverán a otro. En otros casos si tenemos que la asociación e permanente el objeto no podrá salir del espacio en el que está
isNavigable	**Boolean**	Indica si la asociación tiene un conocimiento bi-direccional. Se desprende de las asociaciones de UML.
lowerBound, upperBound	**Integer**	Indican la cardinalidad de la relación. Se desprenden las asociaciones de UML

Constraints:

Profile Notation:

Stereotype	**Superclass**	**Keyword**	**Properties**	**Abstract**	**Icon**
ColaborativeAsociation	*Asociation*			*yes*	

Tool	
Super Class:	CollaborativeElement
Descripción:	La herramienta provee una forma de controlar la información compartida por los usuarios. A través de ellas se crean los objetos compartidos (shared objects). Instancias de herramientas son: o Chat

	o Pizarrón compartido o shared blackboard o Editores de textos colaborativos o Graphical tools

Propiedades:

/workspaces	**WorkSpace**	Representa los workspaces en los que se usa o esta colocada la herramienta. `Context Tool def: workspaces = self.colaborativeAssociation.source -> select (p|p.oclIsKindOf(WorkSpace))`

Constraints:

Profile Notation:

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
Tool	*CollaborativeElement*	*tool*		*No*	

LocationRelationship

Super Class:	**ColaborativeAssociation**
Descripción:	Para expresar asociaciones de ubicación entre diferentes elementos colaborativos. Una herramienta puede estar ubicada en un determinado Workspace. También un usuario o rol pueden estar algún espacio. Recordemos que la ubicación de las cosas puede ser permanente o transitoria.

Properties:

Constraints:

Los elementos conectados son instancias de Workspace y Tool o de Workspace y SharedObject

self.source.oclIsKindOf(Workspace) and

(self.target.oclIsKindOf(SharedObject) or self.target.oclIsKindOf(Tool))

Profile Notation:

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
LocationRelationship	*CollaborativeAssociation*	*contains*		*No*	

UseRelationship

Super Class:	**ColaborativeAssociation**
Descripción:	Sirve para expresar relaciones de uso entre diferentes elementos colaborativos. Por ejemplo entre las herramientas que pueden usar los roles o las herramientas que pueden usarse en una sesión.

Properties:

Constraints:

Los elementos conectados son instancias de SharedObject, Tool, Session y CollaborationRole

(self.source.oclIsKindOf(CollaborationRole) or self.source.oclIsKindOf(Session))

and

(self.source.oclIsKindOf(Tool) or self.source.oclIsKindOf(SharedObject))

Profile Notation:

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
UseRelationship	*CollaborativeAssociation*	*use*		*No*	

ParticipationRelationship

Super Class:	ColaborativeAssociation
Descripción:	Expresa la participación de un rol en una sesión.

Properties:

Constraints:

Los elementos conectados son instancias de Session y CollaborationRole

self.source.oclIsKindOf(Session) and self.target.oclIsKindOf(CollaborationRole)

Profile Notation:

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
ParticipationRelationship	*CollaborativeAssociation*	*participation*		*No*	

SessionRelationship

Super Class:	ColaborativeAssociation
Descripción:	Las relaciones entre los espacios de trabajo y las sesiones están modeladas como una clase de la jerarquía encabezada por la clase abstracta *SessionRelationship*. Existen dos tipos de relaciones básicas para la contextualización de sesiones: La relación de dependencia (DependenceRelationship) representa la pertenencia de la sesión en dicho espacio o contexto. De esta forma todas las sesiones que pertenezcan a un workspace están contextualizadas y por lo tanto podrán tener acceso a las sesiones compartidas que hagan referencia a ese espacio. La relación comparte (*ReferenceRelationship*) representa la posibilidad de que usuarios en una sesión perteneciente al mismo espacio de una sesión compartida tengan acceso a dicha sesión. Una sesión compartida es instanciada una única

	vez durante toda la ejecución de la aplicación Groupware. Por lo tanto, al ser activada una instancia de sesión en un espacio, activarán simultáneamente a las respectivas sesiones compartidas del mismo.

Properties:

Constraints:

Los elementos conectados son instancias de Session y Workspace

self.source.oclIsKindOf(Workspace) and self.target.oclIsKindOf(Session)

Profile Notation:

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
SessionRelationship	*Association*			*Yes*	
ReferenceRelationship	*SessionRelationship*	*reference*		*No*	
DependenceRelationship	*SessionRelationship*			*No*	

9.3.1.3. Protocols

El paquete Protocols contiene los elementos principales de los procesos colaborativos como transiciones y estados. Su objetivo principal es definir el proceso social que se lleva a cabo. Los protocolos son usados para modelar, soportar, guiar y estructurar el proceso social. Los protocolos definen qué herramientas y artefactos pueden ser utilizados por qué rol o por que usuario. Los elementos de este paquete quedan descritos en el metamodelo que podemos apreciar en la Ilustración 16. Se crearon clases específicas que extienden las clases de UML que sirven para representar las máquinas de estado.

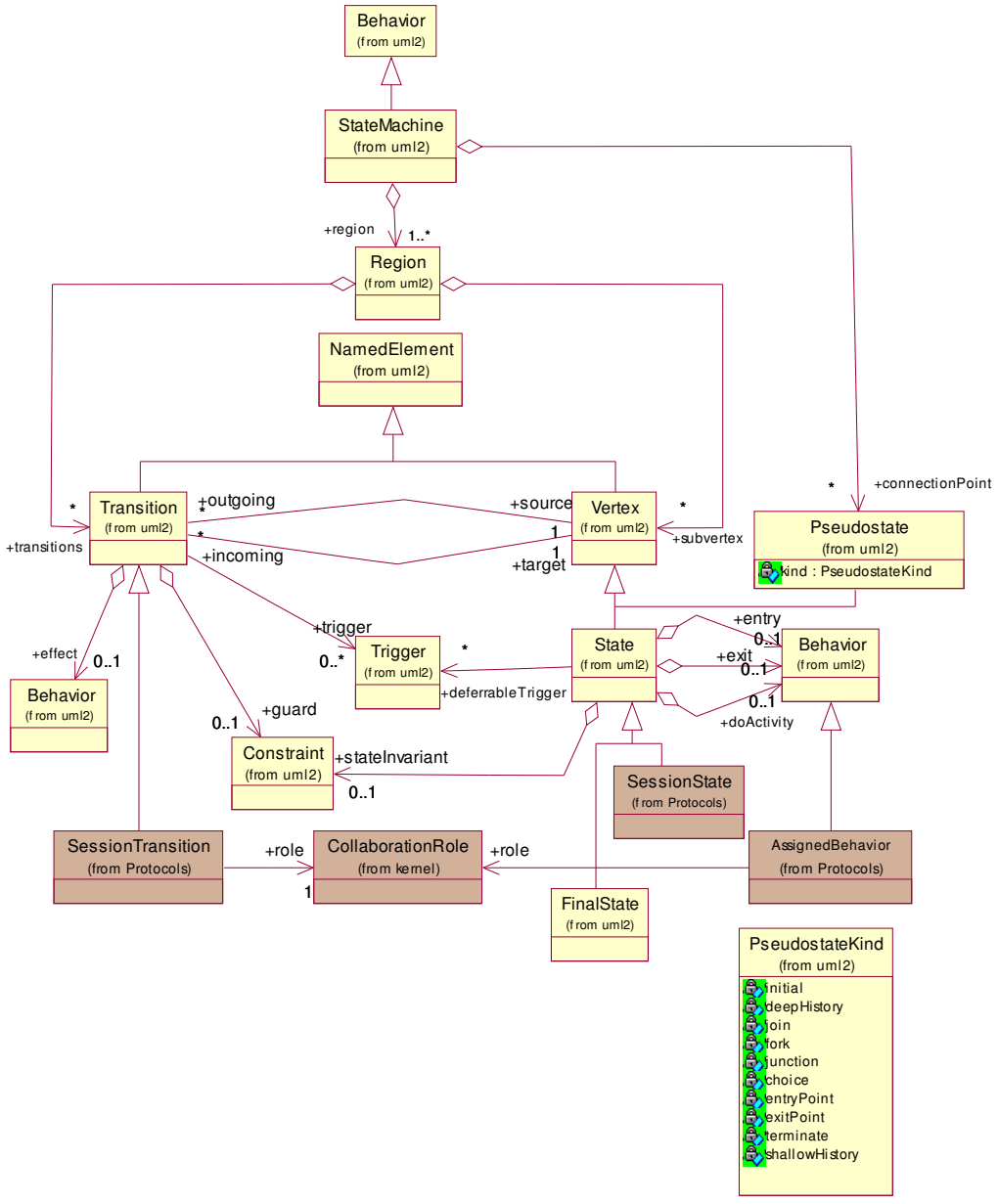


Ilustración 16: Diagrama de clases del paquete Protocols

Session Transition	
Super Class:	**Transition**
Descripción:	Una transición representa el paso de un estado a otro cuando se

	cumplen determinadas condiciones.	
Properties:		
role	**CollaborationRole**	Representa al rol que ejecuta la transición.
Constraints:		

Profile Notation:

Stereotype	Superclass	Keyword	Properties	Abstract	Icon
SessionTransition	*Transition*		*role*	*No*	

SessionState		
Super Class:	**State**	
Descripción:	Es una condición en la que puede estar una sesión en algún momento de su ciclo de vida, durante un cierto tiempo. Esta condición generalmente es un conjunto de acciones que pueden ser desplegadas por un rol (por ej., el orador puede ceder la palabra, el alumno puede solicitar la palabra). Al ser una subclase de State de UML puede ser parte de otra máquina de estado que la contenga. De esta forma se pueden generar procesos complejos.	
Propiedades:		
operations	**List**	Representa la lista de operaciones que se van a desarrollar dentro de este estado. En particular pueden ser todas las operaciones que se realizan en la sesión.
Constraints:		

Profile Notation:					
Stereotype	**Superclass**	**Keyword**	**Properties**	**Abstract**	**Icon**
SessionState	*State*			*No*	

AssignedBehavior		
Super Class:	**Behavior**	
Descripción:	Representa una actividad asignada a un rol particular.	
Properties:		
role	**CollaborationRole**	Representa al rol que ejecuta la transición.
Constraints:		

Profile Notation:					
Stereotype	**Superclass**	**Keyword**	**Properties**	**Abstract**	**Icon**
AssignedBehavior	*Behavior*		*role*	*No*	

9.3.2. Ejemplo de diseño

Para ejemplificar la utilización del Lenguaje vamos a plantear un hipotético sistema en el que intervienen diversos roles, varias sesiones distintas con herramientas colaborativas y aparecen distintos espacios donde se realizan las actividades.

Supongamos que tenemos que desarrollar una aplicación colaborativa para que un equipo interdisciplinario pueda desarrollar una aplicación multimedia. A modo de especificación identificamos que aparecerán distintos roles principales como el *lider* del proyecto, que será el encargado de coordinar las actividades del grupo, el *guionista* que será el

responsable de la redacción del guión y finalmente el *programador*, que será el responsable de implementar el montaje.

Como comentamos en la definición de rol, los usuarios podrán cambiar el rol dinámicamente y este cambio será manejado por los protocolos de colaboración. Por ejemplo, el mismo usuario podrá tener el rol de "*lider*" en algunas sesiones pero podrá actuar como "*mediador"*, en algunas situaciones o como "*participante"* en otras. Por ello es que durante el transcurso del diseño pueden aparecer otros roles que participan en alguna sesión en particular.

En el caso particular de las herramientas colaborativas asumimos que los usuarios podrán acceder a un repositorio de información compartida, que llamaremos repositorio compartido donde colocarán los recursos multimediales, una herramienta de comunicación de tipo Chat; un editor de texto colaborativo; una herramienta como para hacer una presentación, que llamaremos Conference y una herramienta para construir la presentación la presentación multimedia; entre otras. Se asume que las herramientas colaborativas ya están desarrolladas y que importarán en las etapas de diseño correspondientes.

Por otro lado, en el lenguaje propuesto CSSL, al ser compatible con UML, ya que se aprovechan los recursos que UML provee. En este sentido es que encontramos de utilidad el diagrama de Casos de Uso para modelar las responsabilidades de los roles. Una característica fundamental es que varios actores trabajen en un mismo caso de uso y notamos que el concepto de casos de uso de UML no contradice esta posibilidad. En la Ilustración 17 se muestra un diagrama de casos de uso para el ejemplo propuesto.

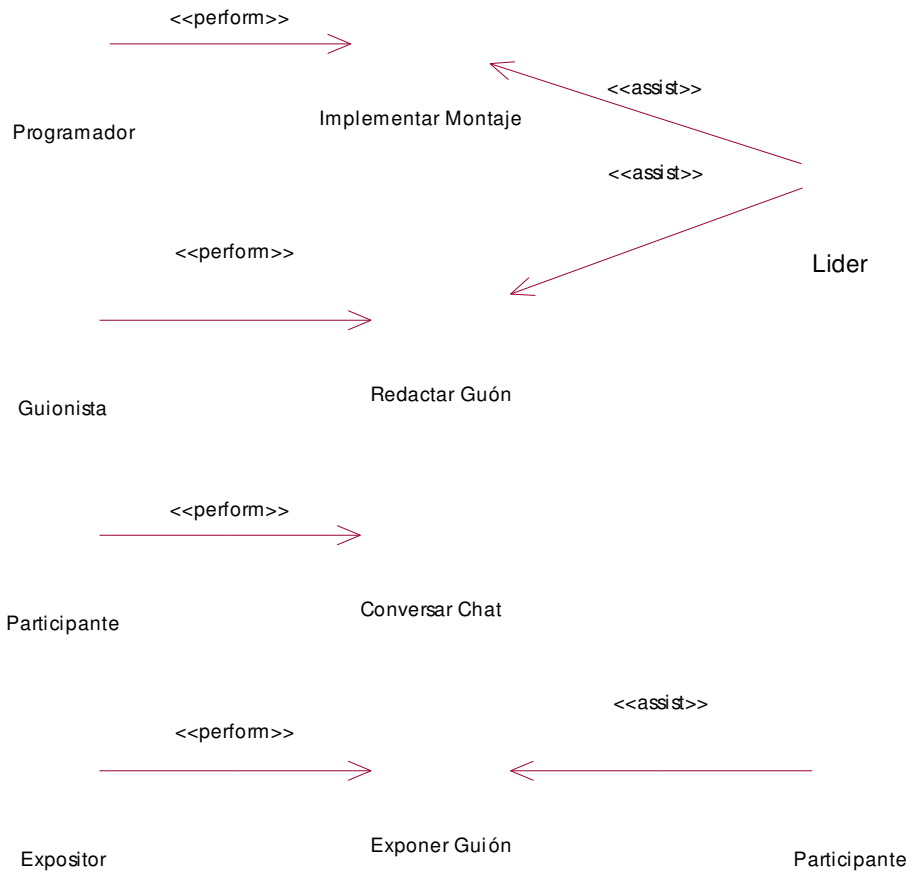


Ilustración 17: Casos de Uso

Es importante destacar que en relación a la derivación automática de modelos, los casos de uso no permiten obtener un modelo derivado, pero esto tampoco ocurre en UML clásico. Para los sistemas colaborativos los diagramas de casos de uso servirán como primera aproximación al análisis del sistema. Aunque no haya una derivación automática, en general se va a entender que los actores se transformarán en roles y los casos serán sesiones.

Siguiendo con nuestro ejemplo, en las próximas ilustraciones veremos distintos diagramas del sistema de creación y presentación de una aplicación multimedia. Los distintos

diagramas son porciones del diseño completo. Elegimos mostrar los diagramas centrándonos en las sesiones que se pueden realizar dentro de los espacios y se representan en los siguientes diagramas: Ilustración 18, Ilustración 19, Ilustración 20, Ilustración 22.

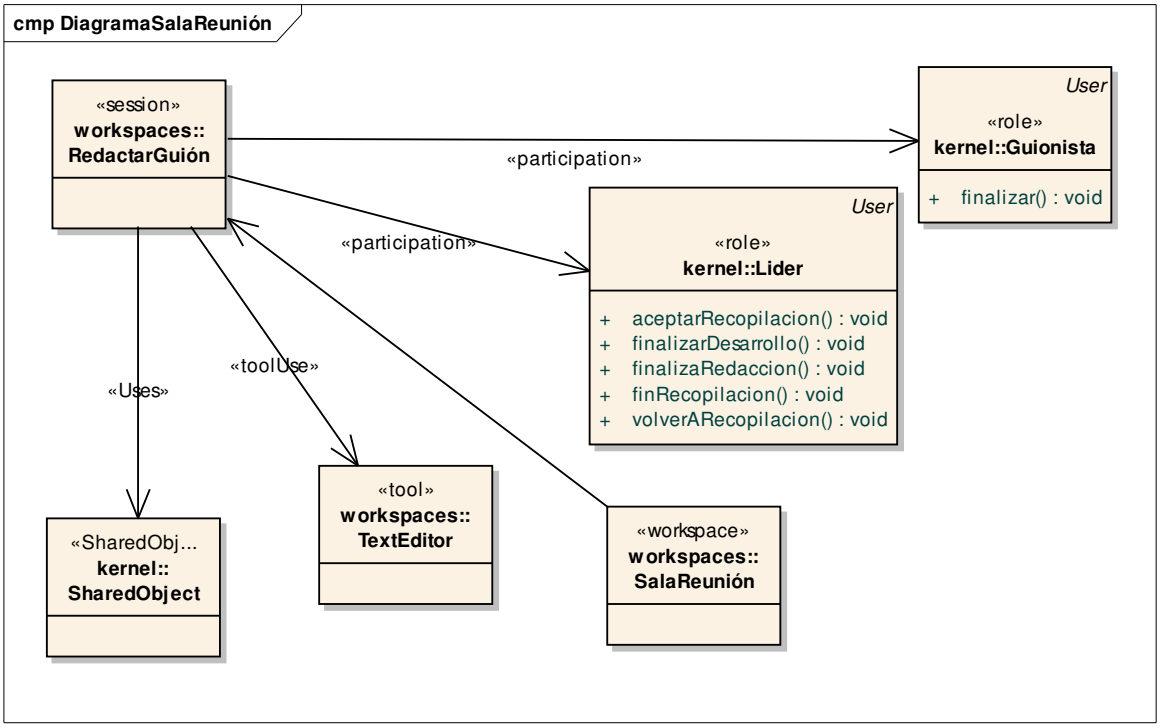


Ilustración 18: Diagrama de la sala de reunión

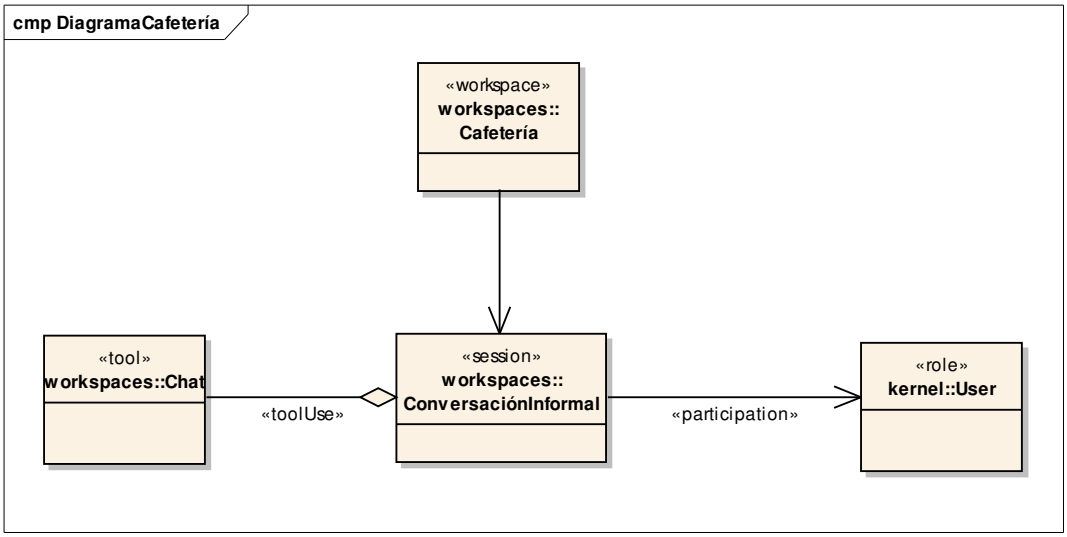


Ilustración 19: Modelo de la sala cafetería

En la Ilustración 20 y la Ilustración 21 se ven sintaxis concretas alternativas para la misma representación de la sala de capacitación. Al final de este capítulo se presentará otra sintaxis concreta basada en formularios, orientada a simplificar la especificación de sistemas colaborativos para usuarios no acostumbrados a realizar diseños con Objetos.

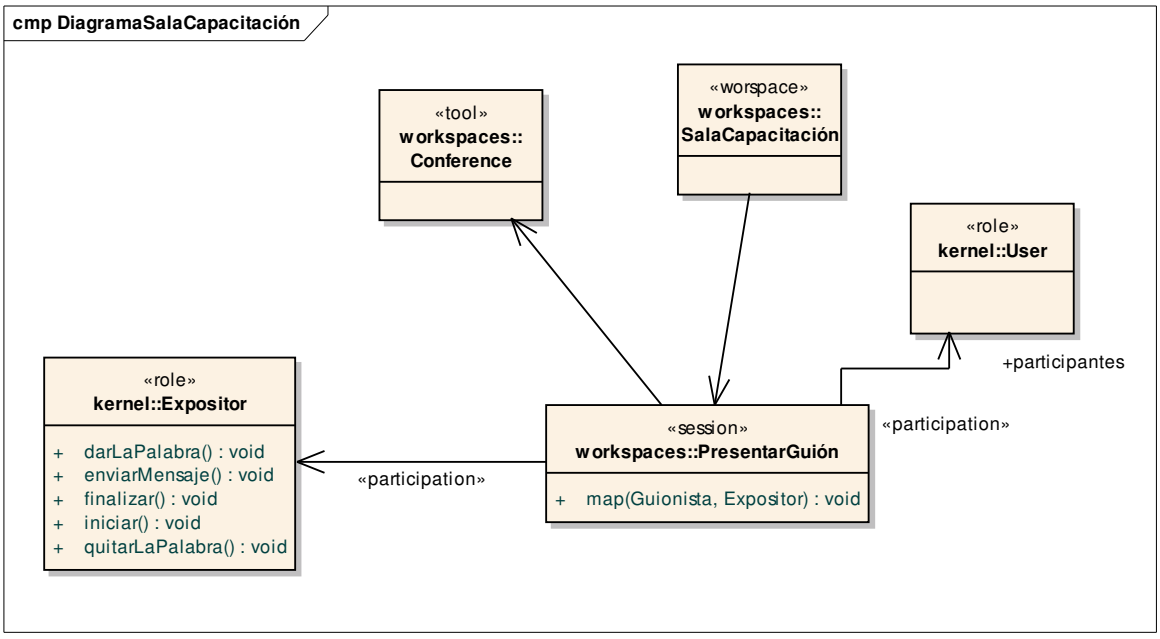


Ilustración 20: Modelo de la sala de capacitación

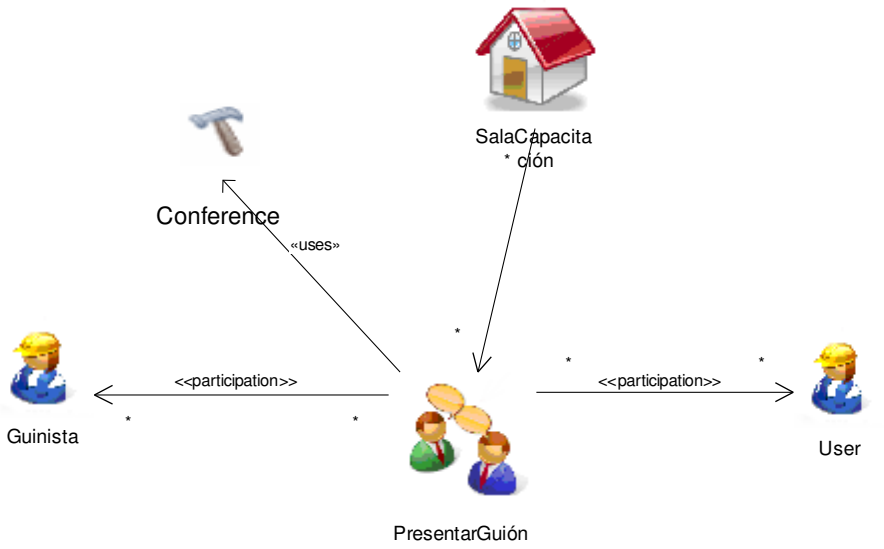


Ilustración 21: Modelo de la sala de capacitación alternativo

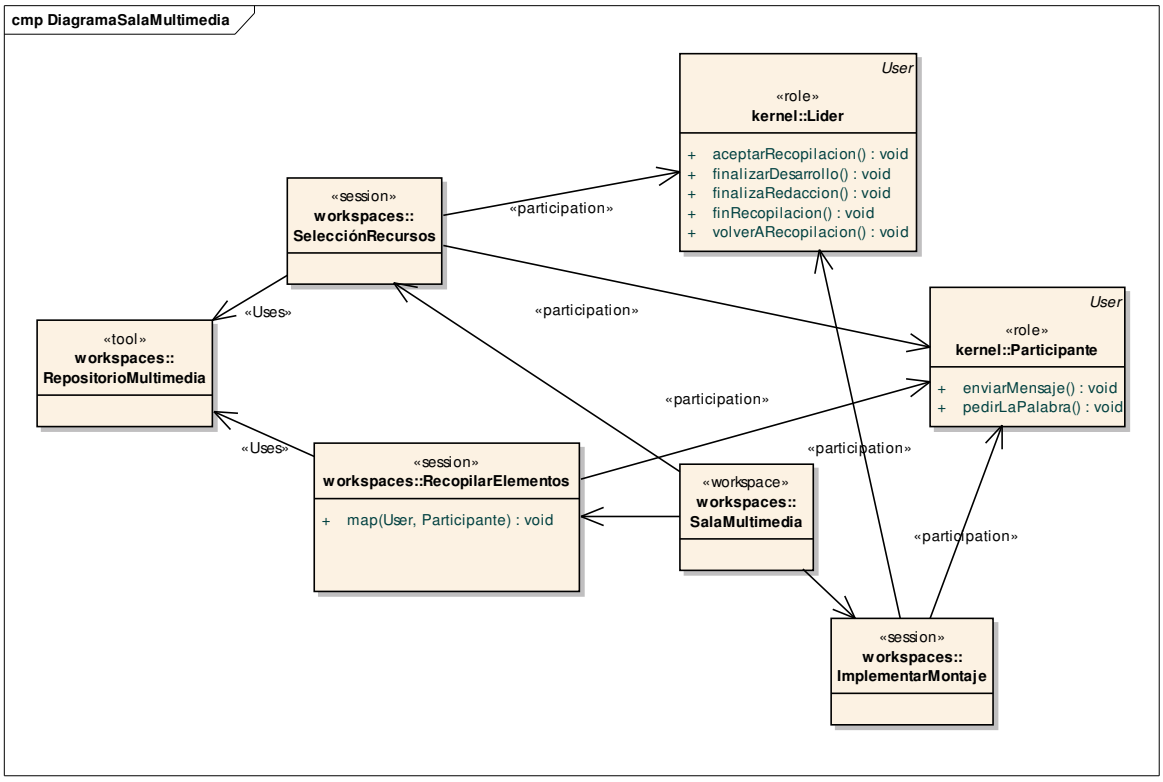


Ilustración 22: Diagrama de la sala de multimedia

Continuando con en ejemplo en la siguiente ilustración identificamos los roles que intervendrán en las distintas sesiones. Además de los roles inicialmente enunciados, como lider, guionista y programador aparecen otros roles como expositor y participante. En este ejemplo queremos mostrar que pueden aparecer roles que son utilizados en una sesión en particular; en este caso el *Expositor* tiene participación en la sesión PresentarGuión. En el diseño se detallará que rol se convertirá en el expositor, nosotros elegimos al Guionista como el rol que funcionará como expositor. Los otros usuarios funcionarán como participantes de dicha sesión.

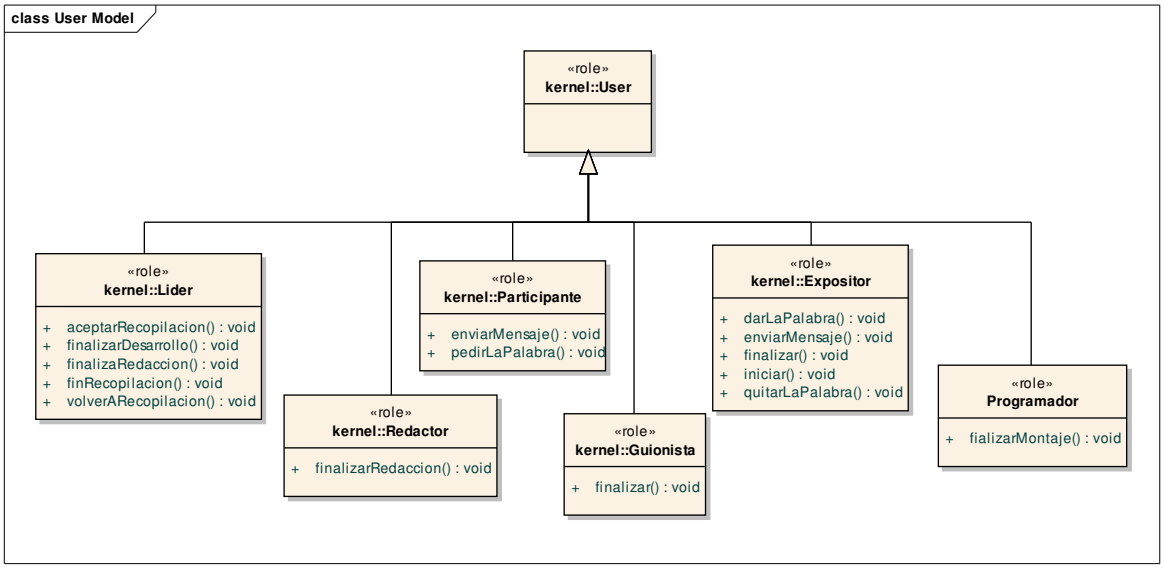


Ilustración 23: Modelo de roles

Por lo que vimos hasta ahora, utilizamos elementos del lenguaje definidos en los paquetes *Kernel* y *Workspace*. De alguna manera diseñamos las características del entorno colaborativo en función de los espacios, que herramientas se van a usar, que roles intervienen y en que sesiones colaborativas habrá. Pero la descripción de una aplicación colaborativa involucra detallar algunos procesos sociales que llamamos protocolos de colaboración. En esta parte se modela la interacción entre los roles, indicando que cosa pueden hacer en cada momento. En el lenguaje CSSL se definió el paquete *Protocol* para

modelar este tipo de comportamiento dinámico de los sistemas colaborativos. Se eligió una notación similar a los diagramas de máquinas de estado de UML. En la Ilustración 24 vemos diagramado el proceso colaborativo que guiará a los usuarios en la construcción del sistema multimedial que se plantea en el ejemplo.

Como en las maquinas de estado, los estados pueden contener a su vez una máquina de estado, generando estructuras de comportamiento anidadas. En la Ilustración 24 se muestra el diseño de un protocolo de alto nivel en el que los estados son sesiones. Dentro de cada sesión habrá un protocolo más detallado como el que se muestra en la Ilustración 25, donde los estados son las operaciones que puede realizar cada rol en un determinado momento.

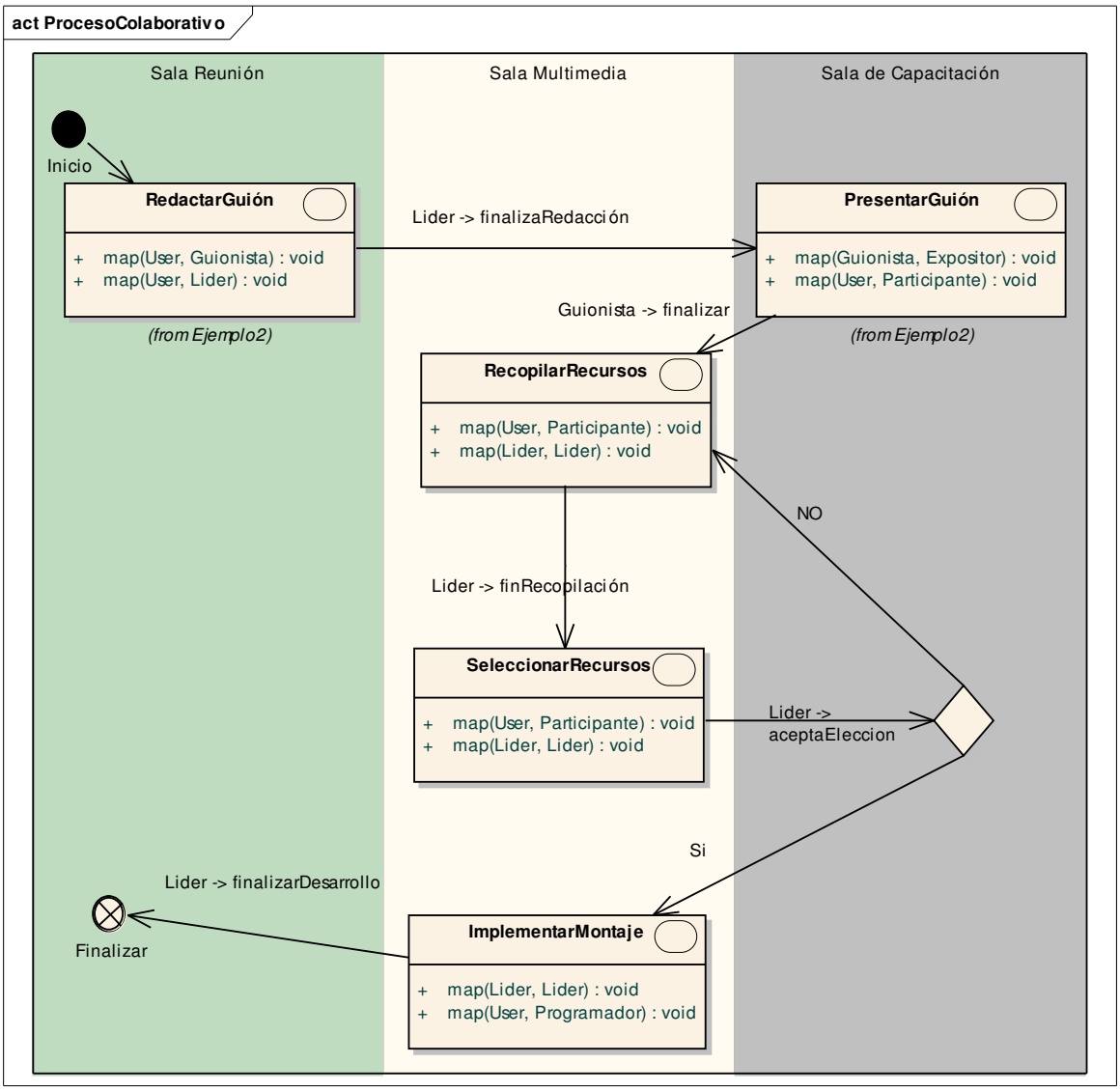


Ilustración 24: Diagrama de proceso colaborativo

9.3.2.1. Especificación de las sesiones de colaboración.

Acá se detalla la participación de cada usuario dentro de una sesión. Cada estado representa un conjunto de operación que los roles pueden realizar. De esta forma se puede detallar que operaciones puede realizar cada rol dentro de cada sesión y se modela la interacción entre los roles. En el ejemplo vemos que en un estado el participante va a pedir la palabra y que va a poder enviar un mensaje si el expositor le otorga la palabra.

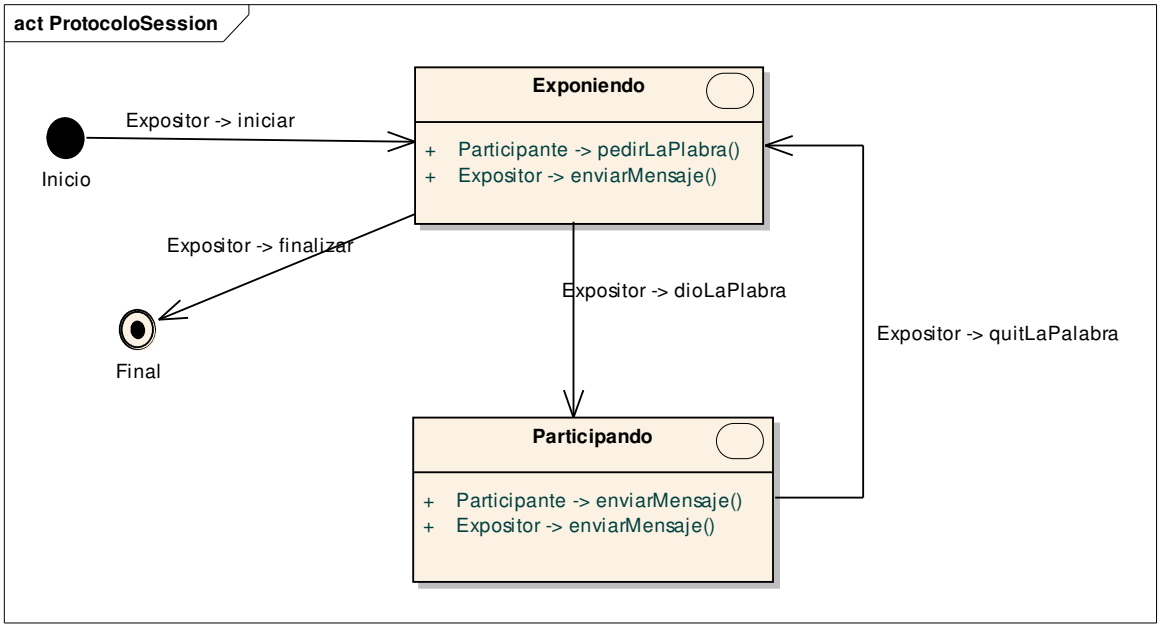


Ilustración 25: Diagrama de protocolo de sesión

9.3.2.2. Sintaxis abstracta y concreta

Como vimos en la sección anterior, el metamodelo define los conceptos del dominio que se utilizarán en el lenguaje. Lo que en los niveles de abstracción de la OMG [] se encuentra en el nivel M2. Y como mencionamos, la responsabilidad primaria de la capa del metamodelo es definir un lenguaje para especificar modelos. En definitiva el metamodelo define la sintaxis abstracta, a partir de la cual se pueden obtener más de una sintaxis concreta. La sintaxis abstracta suele no ser suficiente. Se necesita una sintaxis concreta que provea una representación visual de los modelos. Cada lenguaje de modelado tiene una forma de representación junto con una notación. La forma de representación de la mayoría de los lenguajes es una combinación de gráficos con algo de texto. Los lenguajes de modelado también pueden estar basados en otras representaciones, como matrices, tablas, formularios o puramente textual. La instanciación del metamodelo (es decir la obtención de un modelo conocido como el Nivel M1 del los niveles de abstracción de la OMG) es una forma de obtener la sintaxis concreta a partir del metamodelo.

A partir de un metamodelo se puede definir más de una sintaxis concreta. En este trabajo presentamos otra sintaxis concreta a partir del metamodelo que permitirá a personas menos relacionadas con el diseño orientado a objetos definir las características de los sistemas colaborativos. En un trabajo de tesis que se comentará en próximas secciones los diseñadores podrán trabajar en forma indistinta con una sintaxis o la otra.

La sintaxis alternativa se compone de un conjunto de formularios que permiten definir los elementos que componen un sistema colaborativo.

Se describen a continuación las fichas que podrían se utilizadas por los especialistas para diseñar la aplicación colaborativa. Cada Ficha tiene relación con elementos del modelo.

Ficha de Rol	
Información General:	
Nombre:	// Ingrese el nombre del Rol
Descripción:	// Describir el Rol
Propósito/objetivo:	// Explicar para que se va a usar. Plantear que objetivos tiene. Que tipo de actividades que tipo de actividades realiza
Información Detallada:	

Responsabilidades	// Liste las actividades que desarrolla este rol
	// Por ejemplo:
	Enviar mensajes (con el foro)
	Pedir la palabra

Herramientas:	// Liste las herramientas que se usarán en esta sesión.
	// Por Ejemplo:
	Video-conferencia
	Editor de Texto colaborativo
	Chat

Objetos compartidos	// Liste los objetos que se van editar en esta sesión // Tenga en cuenta con que herramienta se va a trabajar.

Ficha de Worksapce	
Información General:	

Nombre:	// Ingrese el nombre del Workspace
Descripción:	// Describir el Espacio Acá hay que explicar como se va a usar este espacio colaborativo. Plantear que objetivos tiene. Que tipo de actividades se pueden desarrollar dentro de él
Información Detallada:	

Sesiones	// Liste las sesiones que se desarrollarán en este espacio (no es necesario nombrar las herramientas. Solo se tiene en cuenta las actividades que se realizan)
	// Por ejemplo:
	Clase
	Discusión
	Consulta

Herramientas:	// Liste las herramientas que se usarán en este espacio
	// Por Ejemplo:
	Video-conferencia
	Editor de Texto colaborativo
	Chat

Objetos compartidos	// Liste los objetos que se van editar en este espacio // Tenga en cuenta con que herramienta se va a trabajar.

Ficha de Sesión	
Información General:	
Nombre:	// Ingrese el nombre de la Sesión
Descripción:	// Describir la sesión
Propósito:	// Explicar para que se va a usar. Plantear que objetivos tiene. Que tipo de actividades se pueden desarrollar dentro de él. Llegar al propósito puede ocasionar fin la finalización de la Sesión.
Información Detallada:	

Roles participantes	// Liste los roles que intervienen en esta sesión
	// Por ejemplo:
	Mediador
	Participante

Herramientas:	// Liste las herramientas que se usarán en esta sesión.
	// Por Ejemplo:
	Video-conferencia
	Editor de Texto colaborativo
	Chat

Objetos compartidos	// Liste los objetos que se van editar en esta sesión // Tenga en cuenta con que herramienta se va a trabajar.

Siguiendo el ejemplo anterior escribiremos algunas fichas que describirán la forma de utilizar esta sintaxis concreta.

Ficha de Rol	
Información General:	
Nombre:	Lider
Descripción:	Encargado de coordinar las actividades del grupo
Propósito/objetivo:	El objetivo de este rol es el de acompañar la evolución del trabajo en las distintas sesiones. Participa en la mayoría de las sesiones y es el que aprueba y desaprueba el trabajo en cada instancia
Información Detallada:	
Responsabilidades	Se listan las acciones que desarrolla este rol en las distintas sesiones • aceptaRecopilación: Acepta el trabajo de recopilación • finalizaDesarrollo: Finaliza la sesión de Desarrollo y se pasará a la próxima sesión en el proceso colaborativo. • finalizaRedacción: Finaliza la sesión de Redacción del Guión • finalizaRecopilación: Finaliza la sesión de Recopilación de material que se utilizará en la implementación del Montaje • volverARecopilación: Vuelve a la sesión de recopilación de materiales.
Herramientas:	TextEditor: Herramienta para escribir en forma colaborativa Chat: Herramienta para comunicarse
Objetos compartidos	Guión: documento que se completa por los distintos roles

Ficha de Worksapce	
Información General:	
Nombre:	SalaMultimedia
Descripción:	En este espacio se trabajan con herramientas específicas para la selección de material multimedia que luego se utilizarán en la implementación del montaje.
Información Detallada:	
Sesiones	• Selección de Recursos • Implementar Montaje • Recopilar Elementos
Herramientas:	• Repositorio de recursos Multimediales • Chat • Vooting tool
Objetos compartidos	• Objeto multimedia del repositorio • Guión • Log del Chat

Ficha de Sesión	
Información General:	
Nombre:	PresentarGuión
Descripción:	En esta sesión un usuario hace una presentación y al final otros usuarios participantes pueden hacer preguntas sobre la presentación realizada.
Propósito:	El guionista será el presentador y el resto de los roles será los participantes de la misma. El expositor utilizará la herramienta de Conference para realizar la presentación y los participantes podrán pedir la palabra para realizar alguna pregunta con la herramienta Chat.

Información Detallada:	
Roles participantes	• Expositor • Participante
Herramientas:	• Conference: Esta herramienta le permite al expositor compartir información con los otros participantes y guiarlos en la presentación del guión • Chat
Objetos compartidos	• La presentación • Log del Chat

En definitiva el CSSL nos provee una sintaxis abstracta que nos permite representar los aspectos relevantes de los sistemas colaborativos. A partir de la sintaxis abstracta se obtienen distintas sintaxis concretas y a través de ellas, que se presentaron en este capítulo, se pueden describir los aspectos estructurales (principalmente utilizando los paquetes Kernel y Workspace) y los aspectos dinámicos de los sistemas colaborativos. Estos lenguajes concretos nos permiten representar de forma simple y menos ambigua los elementos del dominio colaborativo (Workspace, Role, Tool, Session…etc.). Asimismo al ser un lenguaje bien formado y compatible con UML permite intercambiar modelos con distintas herramientas de modelado. En la documentación del ejemplo hemos usado por un lado una herramienta de UML y en un trabajo en curso [8] estamos armando un editor específico que permite trabajar con las dos sintaxis concretas (la gráfica y la basada en formularios).

Por otro lado, el diseño de los procesos colaborativos y protocolos también se puede tener una representación textual de los estados y transiciones aunque la representación gráfica es más cómoda y más conocida.

10. Método de especificación

10.1. Introducción:

La intensión de esté método es brindar un conjunto de herramientas de especificación que permitan concebir y especificar ambientes Groupware. Dichas herramientas estarán apoyadas en el lenguaje CSSL definido en el capítulo anterior y brindarán una colaboración importante en el relevamiento y captura de funcionalidad con los usuarios finales. Los modelos obtenidos en este método podrán transformarse luego en modelos de diseño o implementación a través por ejemplo de transformación de modelos.

Una diferencia importante en el desarrollo de software colaborativo y software tradicional es la necesidad de contemplar aspectos que introducen las aplicaciones colaborativas que no son tenidos en cuenta en métodos tradicionales de ingeniería de Software. En la Ingeniería de software tradicional existen numerosos métodos de especificación software (como casos de uso, etc.), pero que desde nuestro punto de vista no cubren todos los aspectos funcionales y no funcionales de las aplicaciones groupware o no están descriptos naturalmente. Una de las razones de ello es que dichos métodos permiten la especificación de aplicaciones en general. Este enfoque en general no permite trabajar con claridad algunos conceptos específicos que aparecen en aplicaciones Groupware con mucha frecuencia y son de altísima relevancia (por ejemplo, sesión, espacio, usuario, rol, awareness, etc.).

Como mencionamos en la sección Justificación del modelo espacial, la utilización de espacios en el diseño de aplicaciones colaborativas colabora en la definición de diferentes estilos de colaboración [64], [65]. Trabajaremos con espacios como *el aula*, una *cafetería* o la *biblioteca*. Estos distintos espacios nos ayudan a entender como los usuarios (o roles)

van a interactuar dentro de los mismos. Este trabajo pretende en definitiva, describir puntualmente este tipo de aplicaciones en función del espacio y como evoluciona el ambiente en función del tiempo y las actividades que realizan los usuarios.

Asimismo tenemos que destacar que existen algunos sistemas de cómputo convencionales que son usados por un grupo de personas pero que no tienen las características de los sistemas colaborativos. Consideremos por ejemplo sistemas que permitan reservar pasajes para una aerolínea. La base de datos de la aerolínea contiene los asientos en los distintos vuelos. Los operadores cambiarán el estado de los asientos reservando los lugares. Más allá de una rudimentaria facilidad de control de acceso, el sistema de reservas no cuenta con ningún soporte de coordinación ni de interdependencia de las actividades de los usuarios que trabajen con el sistema. Este tipo de sistemas son habituales en las empresas (por ejemplo bancos, compañía de seguros, etc.) que contienen estructuras de datos donde los distintos usuarios acceden a los datos desde distintos lugares. Sin embargo, esa funcionalidad, un conjunto de usuarios accediendo a un conjunto de datos, ya es considerada como una funcionalidad habitual donde existen un conjunto de herramientas de diseño conocidas en la comunidad de ingeniería de software.

En este trabajo pretendemos ir un poco más allá y pensar en los requerimientos de sistemas que contemplen la interdependencia de las actividades de los usuarios. Es decir la coordinación y la articulación de las actividades de los distintos actores del sistema. Para poner un ejemplo pensemos en una sesión de trabajo a distancia donde un usuario está explicando algo, otro usuario propone un cambio en la agenda de la reunión. Este cambio de agenda es ahora el objeto sobre el cual el conjunto de usuarios está trabajando y una vez resuelto este tema se puede volver al objeto original de la reunión. En definitiva la articulación del trabajo es una actividad que puede realizarse en forma colaborativa y que en muchos casos requiere herramientas específicas.

Como mencionamos en los capítulos anteriores, existen numerosas características de las aplicaciones colaborativas a tener en cuenta a la hora de pensar en construirlas. Las

aplicaciones son bien diferentes dependiendo de muchos factores, entre ellos se encuentran:

- Distribución de las actividades en tiempo y espacio
- Número de participantes
- Complejidad del trabajo
- Grado de especialidad (o disparidad del grado) de los participantes

Cuanto más distribuidas sean las actividades mas complejo es la coordinación de actividades de los participantes. Asimismo, si el grado de complejidad del trabajo es relativamente bajo, puede articularse razonablemente el trabajo definiendo modos de colaboración. También es importante notar que es mas fácil coordinar las actividades de los participantes cuando los mismos tienen un grado de especialidad homogéneo, ya que utilizan un lenguaje común como se menciona en [61].

La coordinación de actividades introduce un requerimiento adicional a los sistemas colaborativos, que deberán brindar a los participantes de la colaboración un suporte de awareness, un entendimiento de las actividades que realiza y como influyen sobre las actividades de los otros usuarios, mantener información de las actividades pasadas, presentes, planificadas y manejar el trabajo propio de los usuarios, su comunicación, sus gestos, su estado etc. Toda esta interacción deberá ser combinada dinámicamente de manera natural para que los participantes puedan colaborar.

El método propuesto colabora en el entendimiento de las características mencionadas y brinda herramientas para interactuar con los usuarios finales de aplicaciones groupware y permite a un grupo de desarrolladores comunicarse, discutir y elaborar sistemas colaborativos de una forma integral.

10.2. Descripción general del método

El método propuesto, al igual que el **Rational Unified Process** (RUP) de Rational (IBM), se caracteriza por estar dividido en etapas que se pueden desarrollar en forma iterativa e incremental. Las distintas etapas tienen un objetivo de diseño diferente.

Las etapas son las siguientes:

- Describir el entorno colaborativo
- Describir los procesos colaborativos
- Describir los protocolos de colaboración

Antes de empezar a describir las distintas etapas se detallará con que elementos del lenguaje se trabaja en cada una de las etapas.

En la descripción del entorno colaborativo se describirán los roles que intervendrán en el sistema, los espacios donde se van a producir las actividades colaborativas (workspaces) y las sesiones (sessions) que se van a llevar a cabo y que relación tienen que los espacios colaborativos.

En la descripción de los procesos colaborativos se detallan las actividades que los roles desarrollan y que les permiten participar en las sesiones colaborativas. En definitiva, a través de estas acciones los usuarios activarán y finalizarán sesiones colaborativas.

En la etapa de descripción de los protocolos de colaboración se detallan las actividades que los roles tienen dentro de una sesión de colaboración. Estos protocolos detallan que acciones puede realizar el usuario (o el rol) en cada momento.

Recordemos que estas etapas, como en otros métodos, pueden ser desarrolladas en forma iterativa e incremental. Es decir que el diseñador puede descubrir en la etapa de descripción de procesos colaborativos que necesita una sesión que no estaba definida. Esto

hará de alguna manera que se vuelva sobre la etapa que describía en el entorno para detallar la nueva sesión, para luego continuar con la descripción del proceso.

10.2.1. Descripción del entorno colaborativo

La utilización de un diseño del entorno permite especificar el comportamiento de los espacios donde se van a efectuar distintas actividades colaborativas. En este trabajo proponemos realizar un diseño espacial que refleje los distintos estilos de colaboración que se pueden requerir de la aplicación. El lugar se define a partir de la definición de sesiones colaborativas, roles y el vínculo que hay entre ellos. De alguna manera brindarán un contexto a las actividades que se realicen en él. Por ejemplo si pensamos en un espacio *Aula* donde puede participar un rol docente y otro alumno con una herramienta de pizarra compartida seguramente tendrá un estilo de colaboración distinta que en otro espacio Cafetería donde los usuarios podrán conversar usando un *Chat*.

El diseño de espacios físicos y como la gente los usa, es un campo bien conocido en áreas como planificación urbano y arquitectura. Por otro lado, los espacios ayudan a comprender la ubicación y el movimiento de los usuarios (o navegación dentro del sistema). Suponiendo que los sistemas colaborativos tengan a su vez facilidades para que los usuarios naveguen en los ambientes, destacamos que hay numerosas investigaciones y metodologías que nos ayudan a construir ambientes virtuales (como OOHDM [26]).

Particularmente la metáfora de ambiente modela fácilmente que es lo que la gente puede hacer en ese ambiente y es una forma natural de proveer oportunidades colaborativas. Los ambientes, análogamente a los ambientes físicos, pueden ser usados por los grupos dentro de la organización para realizar las actividades grupales. Entre ellos encontramos:

- oficinas personales;
- ambientes compartidos por varios grupos de trabajo, donde se suelen reservar como una sala de conferencia o de capacitación;

- ambientes compartidos dedicados a un proyecto, los cuales suelen ser más pequeños que los anteriores (suelen trabajar entre 3 y 6 personas) y que sirven para que un equipo tenga en él todos sus recursos;
- ambientes compartidos no dedicados a un proyecto, que son similares a los anteriores pero los miembros que lo usan tienen sus propias oficinas y utilizan estos para reuniones eventuales;
- lugares públicos de interacción social, como por ejemplo cafeterías, pasillos o cualquier otra área común;

Estos ambientes físicos son ambientes delimitados, trabajan como contenedores, las cosas tienen una ubicación dentro del ambiente y las personas no pueden estar en más de un ambiente al mismo tiempo.

Se analiza a continuación cada uno de estos ambientes y se relacionarán como se representan en los sistemas colaborativos.

Un ambiente dentro del sistema colaborativo está definido por una colección de workspaces. Un workspace representa un espacio en el que las sesiones se relacionan con él de forma tal de brindar un estilo de colaboración. Recordemos que existen dos tipos de relaciones básicas para la contextualización de sesiones: La relación pertenece se establece entre un espacio de trabajo y una sesión. Representa la pertenencia de la sesión en dicho espacio o contexto. De esta forma todas las sesiones que pertenezcan a un workspace están contextualizadas y por lo tanto podrán tener acceso a las sesiones compartidas que hagan referencia a ese espacio.

La relación comparte también se establece entre un espacio y una sesión. Representa la posibilidad de que usuarios en un espacio compartan la sesión compartida. Una sesión compartida es instanciada una única vez durante toda la ejecución de la aplicación Groupware. Por lo tanto, al ser activada una instancia de sesión en un espacio, activarán simultáneamente a las respectivas sesiones compartidas del mismo. De esta forma las

sesiones compartidas son las que brindan un conjunto de funcionalidad común y contextualizan las actividades que se hacen en dicho espacio.

En los espacios que definiremos también vamos a especificar que roles de usuarios ingresarán a la sesión y como se adaptan al contexto del ambiente

10.2.1.1. Ejemplo de entorno:

Para relacionar las sesiones con los espacios se utiliza la siguiente notación expresada en la Ilustración 28. Las líneas continuas representan la pertenencia de la sesión en un espacio determinado y las de guiones representan la referencia a sesiones por parte del espacio.

El ejemplo lo plantearemos función de una aplicación groupware prototípica llamada "Vital" que fue desarrollada por el grupo de investigación GMD-IPSI en Alemania. [67]

El sistema Vital es un ambiente para aprendizaje colaborativo, preparado para soportar grupos pequeños y medianos de usuarios (entre 2 y 8 usuarios interactuando en forma simultánea). Definen algunos espacios donde se desarrollan las actividades colaborativas como el *Auditorium*, *Discussion Room*, *User Room*… entre otras. Utilizando nuestro lenguaje vamos a describir estos espacios, las sesiones y roles que intervienen en el.

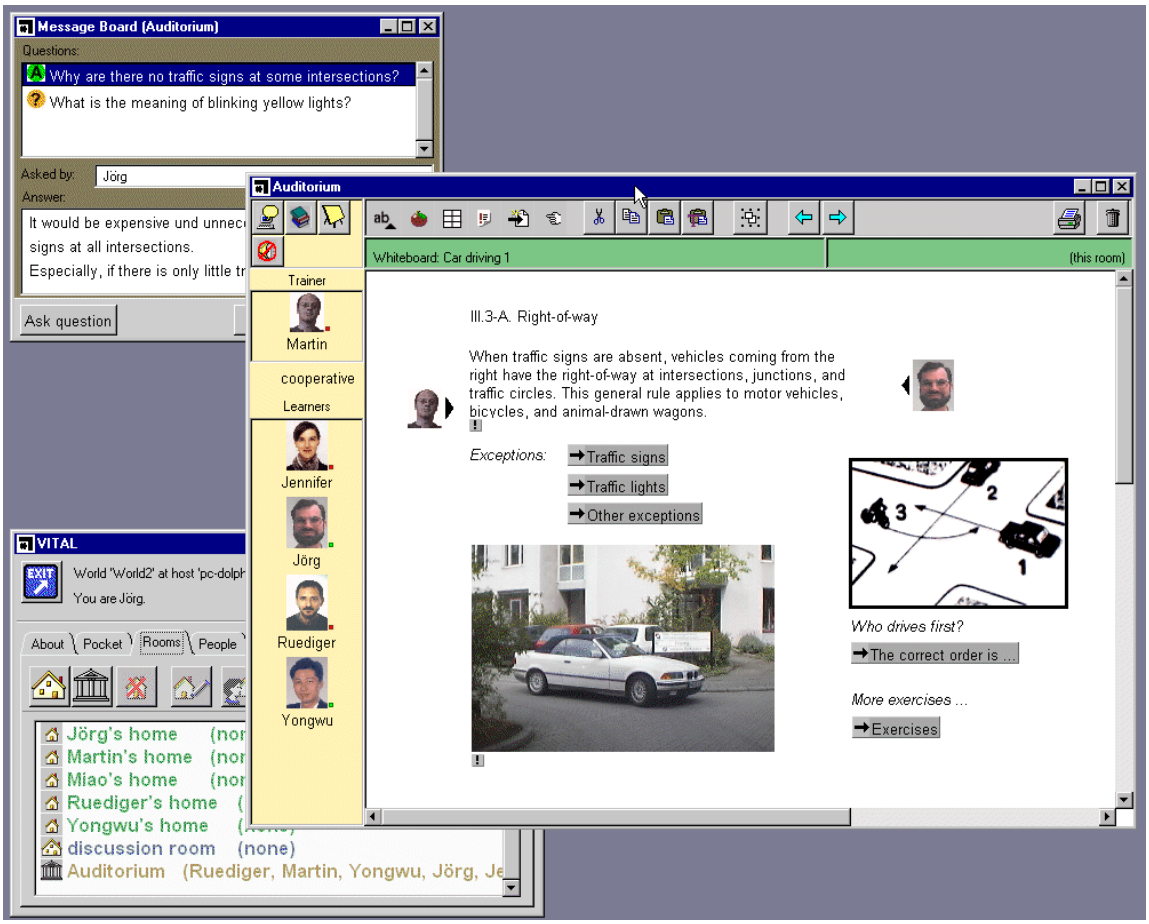


Ilustración 26: Vital; sistema de aprendizaje colaborativo

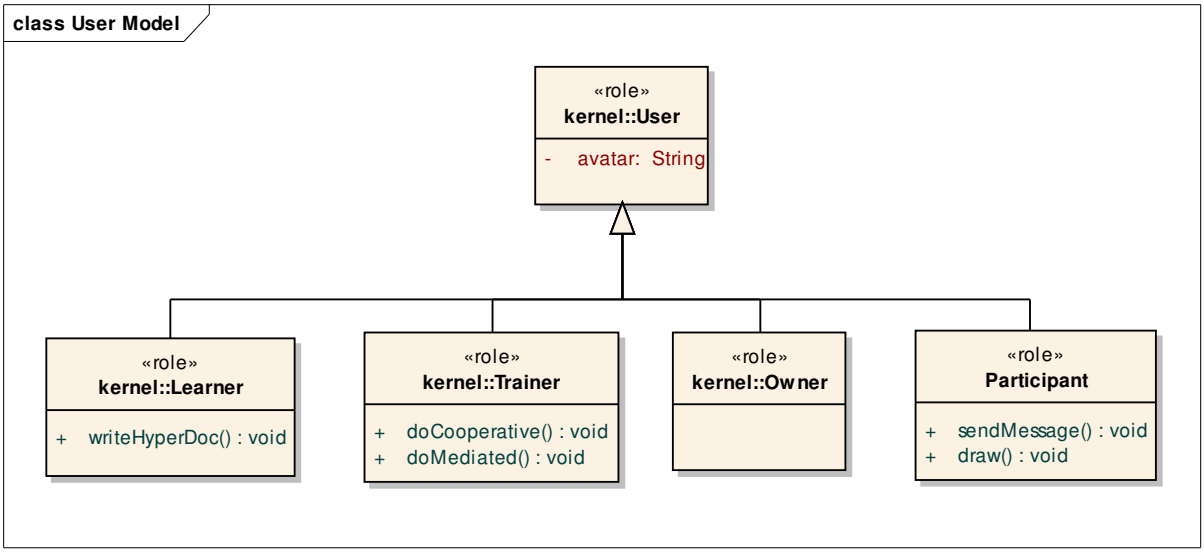


Ilustración 27: Vital; modelo de roles del sistema

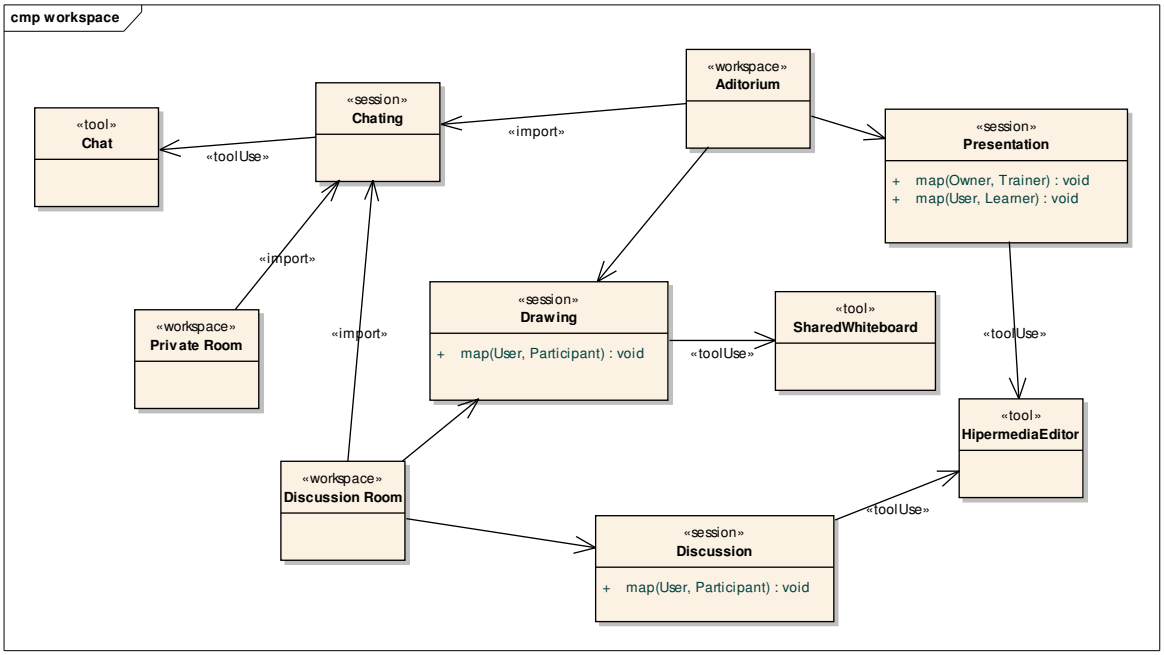


Ilustración 28: Especificación de Entorno Colaborativos (Espacios)

En el ejemplo planteamos 3 espacios dentro una facultad y describimos que sesiones se pueden vincular a dichos espacios. Este ejemplo es una versión simplificada de los posibles espacios que puede tener una Facultad virtual.

10.2.2. Descripción de los procesos colaborativos

A través del protocolo de alto nivel (entre sesiones) se tiene la posibilidad de especificar procesos, que por su complejidad requieren de la intervención de distintos grupos de usuarios (roles) en distintas sesiones de trabajo. Así se pretende especificar la disponibilidad de sesiones dentro del marco de un proceso en el cual se necesitan seguir ciertos pasos para poder ser ejecutado satisfactoriamente. Cada paso será representado por medio de una sesión de trabajo.

Usualmente este tipo de especificaciones está cubierto mediante la especificación de un Workflow. En este trabajo se agregó la asignación y mapeo de roles dinámicamente a medida que el usuario va actuando en las distintas sesiones. Los estados y transiciones del workflow se especifican con la siguiente notación general de la siguiente ilustración.

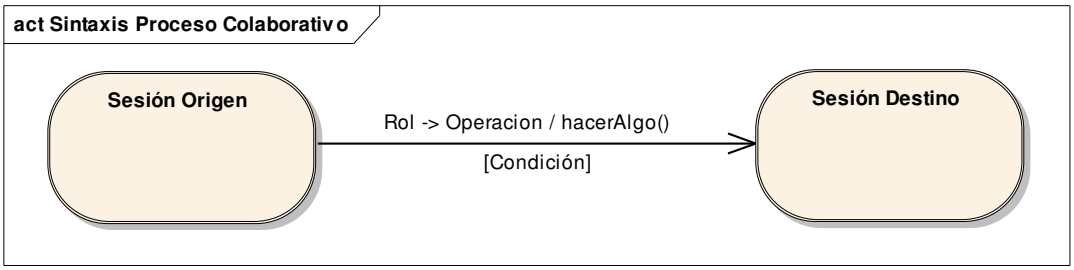


Ilustración 29: Notación general de protocolo entre sesiones

Donde un estado representa la disponibilidad de acceso (Depende del nivel de abstracción del protocolo) que tienen los diferentes roles. Para representar la disponibilidad, se utilizan las funciones acceso. Una función acceso representa la disponibilidad de un rol de acceder a algún elemento (Depende del nivel de abstracción del protocolo).

Así, un estado está compuesto por un conjunto de funciones acceso que definen los permisos de los diferentes roles en un estado determinado del protocolo.

El estado de un protocolo puede cambiar de acuerdo a ciertas reglas. Dichas reglas están representadas por medio de transiciones. Una transición entonces, permite cambiar las posibilidades de acceso de los roles. Para que una transición ocurra se deben satisfacer simultáneamente al menos dos condiciones:

- 1. Ocurrencia de al menos uno de los eventos especificados en la transición.
- 2. Satisfacción de todas las guardas de la transición.

Un evento representa la ejecución de una operación sobre un artefacto por parte de un usuario perteneciente a un rol.

Una guarda representa un conjunto de condiciones en base al estado de las variables que representan el estado actual. El estado de las variables establece una especie de "memoria del global del protocolo" para tomar decisiones.

Un último concepto importante de las transiciones es la ejecución de una secuencia de asignaciones cuando una transición de estados se lleva a cabo. Esta secuencia de asignaciones permite actualizar el estado de las variables que representarán el próximo estado.

10.2.2.1. Ejemplo de proceso colaborativo

El sistema Vital no tiene la posibilidad de definir procesos colaborativos o dicho de otra manera, los usuarios que participan de alguna actividad colaborativa en Vital tienen la libertad de moverse por los distintos espacios y participar en la sesiones colaborativas que quisieran. El grupo del GMD-IPSI incorporó la definición de procesos colaborativos en un sistema que continuó al sistema Vital y que se llama Crocodrile. En Crocodrile se definen procesos de aprendizaje que los usuarios tienen que seguir para aprender determinado tema.

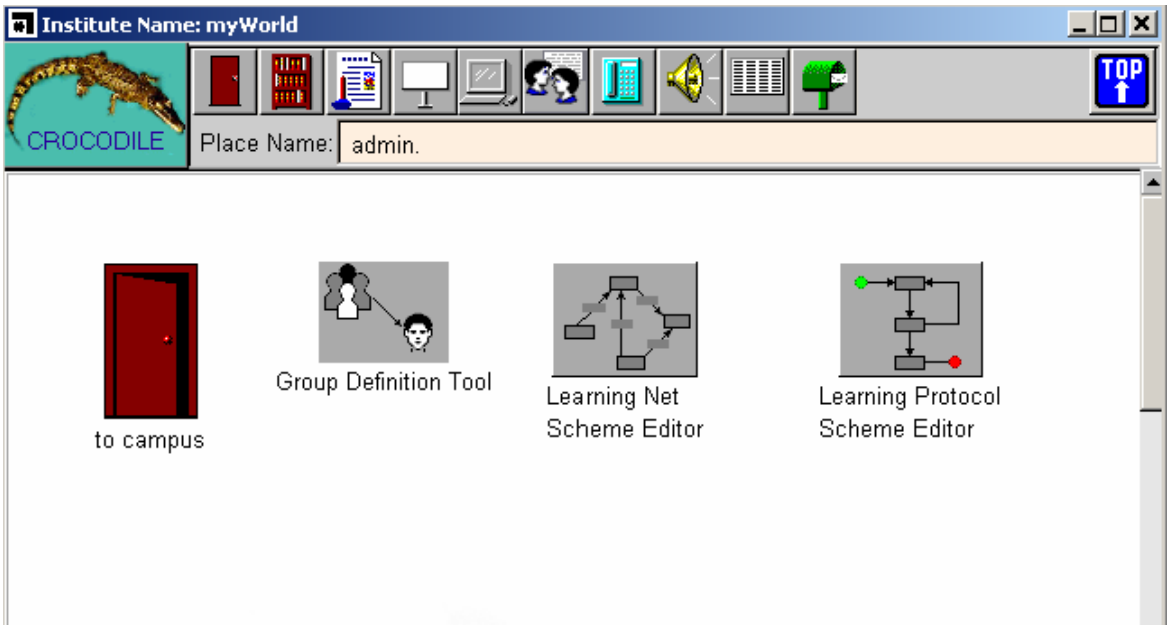


Ilustración 30: Crocodile; modelado de proceso colaborativos

En el contexto de este trabajo vamos a suponer que existe un proceso de enseñanza en el que los usuarios tienen que escribir un documento para presentarlo en el auditorio. Antes de disponerse a escribir el documento los distintos usuarios se reúnen en el discussion room para decidir que tema tiene que preparar cada uno. Luego deberán ir a editar el documento en forma individual para luego realizar una presentación en el auditorio donde participarán todos los usuarios. Durante la discusión en el Discussion Room y la presentación en el Auditorium los usuarios podrán mantener sesiones de comunicación usando una herramienta de pizarra compartida y un chat.

En el ejemplo plantearemos un circuito administrativo de un tratamiento de un expediente en una organización simplificada de consejo académico como se ve en las ilustración 3. El tratamiento de un expediente comienza con la presentación de una nota. Dicha nota es procesada por los empleados de la mesa de entradas. Cuando la nota es confeccionada, se procede a decidir el destino de la nota. Esta discusión se lleva a cabo en una sesión de discusión. Dicha nota puede tener dos destinos: una comisión o el consejo. A partir de ese momento cualquiera sea el camino, se produce la creación de un expediente.

Si el camino que sigue el expediente es el del consejo, se discute el tema en una sesión apropiada y luego se pasa a una votación la que se decidirá pasar a una resolución o a una comisión para analizar el expediente. En cambio si se decide seguir el camino de una comisión. El expediente se discute y se pasa al consejo para que se resuelva sobre el mismo.

Los estados representan la asignación de roles de aplicación a roles dentro de la sesión. Esta asignación toma grupos de usuarios de la aplicación y les asigna tareas específicas del protocolo de la sesión, dichas tareas son definidas en la especificación de las sesiones de colaboración.

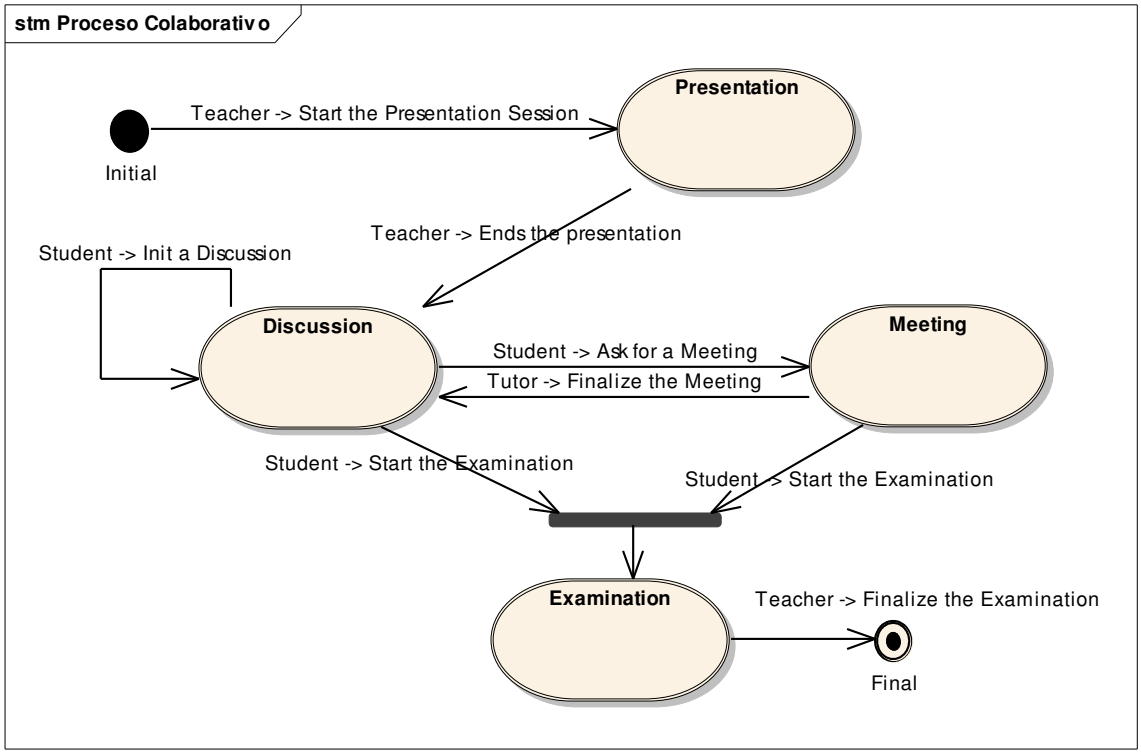


Ilustración 31: Especificación de Procesos Colaborativos

10.2.3. Especificación de las sesiones de colaboración

La especificación dentro de la sesión es la definición de un protocolo que sirve para especificar la sincronización entre los usuarios y los dispositivos de software por medio de los cuales se modifica el modelo de la aplicación.

Por lo tanto, a este nivel, un acceso es representado por una operación perteneciente sobre un artefacto. De esta manera una función de acceso representa la posibilidad de realizar cierta operación sobre un artefacto, por parte de un usuario perteneciente a un rol particular.

Los nombres de los roles en este nivel deben ser descriptivos en función del comportamiento específico dentro del ámbito de la sesión, para ayudar a comprender el comportamiento del mismo.

En cuanto al estado de variables en este nivel podemos decir que cuando una sesión se activa; es decir, un conjunto de usuarios comienzan a ejecutar sus actividades, se crea un espacio de variables en que estas variables se almacenan.

Todo protocolo comienza con un estado especificado, al cual se arriba luego de haber inicializado adecuadamente las variables del espacio de la sesión activa. La notación para expresar protocolos se detalla en forma general como se ilustra a continuación.

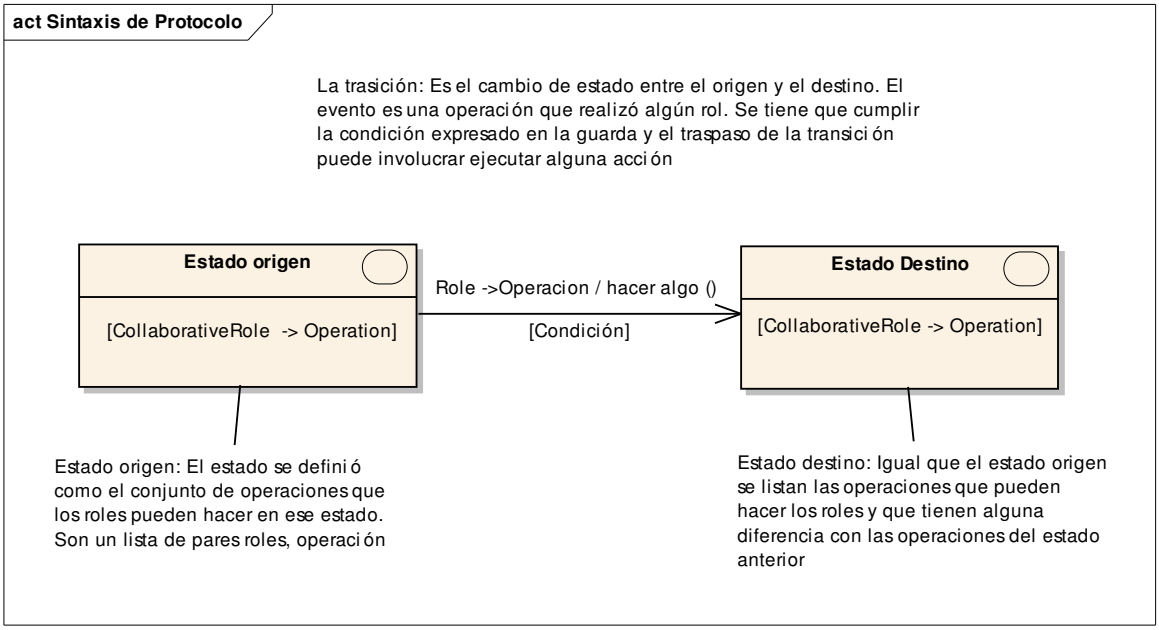


Ilustración 32: Notación general de protocolos estados de una sesión

Las transiciones se tratan de la misma forma que en el caso del protocolo entre sesiones. Además, tienen las mismas características y cumplen la misma función.

10.2.3.1. Ejemplo de protocolo

Para finalizar la concepción de la idea de protocolo se presentará un ejemplo basado en un protocolo de interacción simple denominado protocolo OneTwo. Este protocolo es está basado en la idea del protocolo ProContra (Debate) con ciertas modificaciones.

El objetivo de este protocolo es establecer un protocolo de comunicación en el que dos grupos de participantes (identificados con los roles One y Two) alternen los turnos de contribución a un Chat cuya única operación relevante al protocolo es "*send*". El grupo One debe esperar a que dos participantes del grupo Two ejecuten send en el Chat para poder contribuir nuevamente. En el estado inicial cualquier participante de cualquiera de los grupos puede enviar un mensaje (Chat.send). El estado inicial se representará mediante una S y la transacción se ejecuta cuando comienza la sesión. Esta transición permite

inicializar la variable S que permite representar la cantidad de veces que contribuyeron los participantes del rol Two antes que un participante del rol One.

A este ejemplo se lo denominará como protocolo OneTwoChat.

Como se puede observar en la Ilustración 33 el protocolo comienza en un estado en el que tanto los roles One como Two pueden ejecutar la operación send sobre el Chat. A partir de ese momento existen dos eventos posibles relevantes para el protocolo:

1. Un participante del rol One ejecuta la operación send.
2. Un participante del rol Two ejecuta la operación send.

En ambos casos el en próximo estado deberá asegurar que solo un participante del rol One pueda ejecutar la operación send en el artefacto Chat.

A partir del cambio de transición, en el caso que la contribución haya sido producida por un participante del rol One, la variable continúa con su valor, 0.

En cambio, el valor pasará a 1 en caso que la transición haya sido motivada por un evento de algún participante del rol Two. Así queda descrito el estado del protocolo en la variable S.

En el estado en el que solo un participante del rol Two pueda contribuir existen dos posibilidades. La primera de ellas permite, por medio de la guarda

$$S < 1 \quad (1)$$

que el estado se repita. Si este fuera el caso, la variable S pasa al valor 1. Si observamos con más detenimiento la guarda, indicaría que puede ocurrir otra contribución de un participante de One siempre que antes no haya contribuido.

La segunda, por medio de la guarda

$$S > 0 \quad (2)$$

permite que solo algún participante perteneciente al rol Two pueda ejecutar send en el Chat. Además, si el evento y las guardas permiten que la transición se lleve a cabo, el valor de S pasa a 0.

Otra vez, se puede observar que S mantiene la representación de la cantidad de veces que contribuyeron los participantes del rol Two antes que un participante del rol One El ultimo estado a analizar es el que permite que solo los usuarios pertenecientes al rol One ejecuten Chat.send. Cuando se ejecuta la operación; se pasa al estado en el que la variable S pasa a 0 y solo los participantes del rol Two pueden ejecutar operaciones sobre el Chat. Se observa que en el protocolo queda mismo estado que se tenía cuando un participante de One realizaba la primera operación send en la sesión.

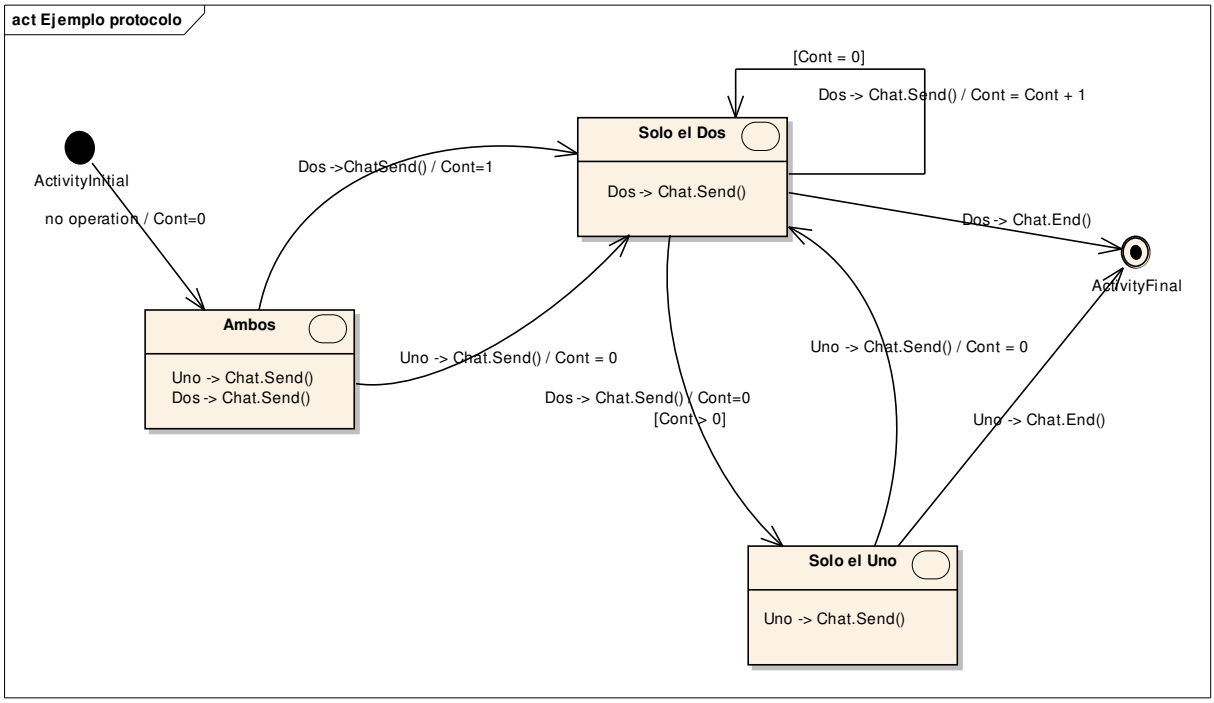


Ilustración 33: Ejemplo de Protocolo OneTwoChat

10.3. Awareness

Una de las características más salientes de los sistemas Groupware es el awareness. El awareness es información que el sistema provee sobre el estado de la colaboración en

relación a la presencia de los usuarios, su ubicación dentro del sistema, nivel de actividad, acciones, cambios, objetos que se utilizan, intención etc. [50], [58]

Esta información facilita el trabajo de los usuarios teniendo percepción o conciencia de lo que está pasando con el y con los otros usuarios en el ambiente colaborativo. Este concepto no es especialmente tenido en cuenta en las metodologías de especificación y diseño tradicionales.

Los niveles de especificación de awareness que se establecerán serán los siguientes:

1. Entorno
2. Espacio de trabajo (workspace)
3. Sesión
4. Artefacto

Cada nivel establece el alcance del awareness en cada elemento. Por lo tanto es posible asociar cualquier tipo de awareness a cualquier nivel. Para poder asociar un tipo de awareness a algún elemento se deben definir estos tipos. El cuadro siguiente describe los tipos más comunes de awareness.

La notación para describir el awareness necesita de la expresión de los elementos anteriormente especificados, así podemos describir el awareness de la forma

Una observación interesante de la Ilustración 34 es que el awareness (con la inclusión de la representación de los artefactos) se expresa en el mismo diagrama de entorno. Por último destacaremos que la inclusión de un nuevo elemento descriptivo, el artefacto, es la mínima unidad de alcance de awareness. En la siguiente tabla se listan los elementos de awareness que se pueden asociar a los distintos niveles de la aplicación

Elemento de Awareness	**Pregunta relevante**
Presence	¿Quiénes están en el entorno, el workspace, la sesión o el

	artefacto?
UserLocation	¿En qué lugar (espacio, sesión, artefacto, etc.) se encuentra un usuario determinado?
Density	¿Que densidad de usuarios en cada lugar del entorno?
UserInformation	¿Qué información conocemos del usuario?
Rol	¿Qué rol tiene el usuario en cada momento?
ActivityLevel	¿Que nivel de actividad tienen los usuarios?
UserAction	¿Qué acción está haciendo el usuario?
PlaceAction	¿Qué acción se está haciendo en un lugar del entorno (espacio, sesión, artefacto, etc.)
UserHistory	¿Cuál es la historia de acciones de un usuario?
Intentions	¿Qué harán los usuarios?
UserBookmarks	¿Cuáles son los bookmarks de un usuario?
Expectations	¿Qué acción se considera que hará luego un usuario considerando el estado actual?
Changes	¿Qué objetos han cambiado?
Objects	¿Cuál es el estado de algún objeto?
Visibility	¿Cuál es la visibilidad de algún usuario? ¿Que objetos puede ver?
Abilities	¿Qué sabe hacer el usuario? Puede tener que ver con el rol
Influence	¿Cual es el campo de influencia del usuario?
ObjectsUsers	¿Qué objetos están siendo usados y por quién?

ObjectHistory	¿Qué cambios se han realizado sobre el objeto y quién lo realizó?

Tabla 2: Elementos de Awareness

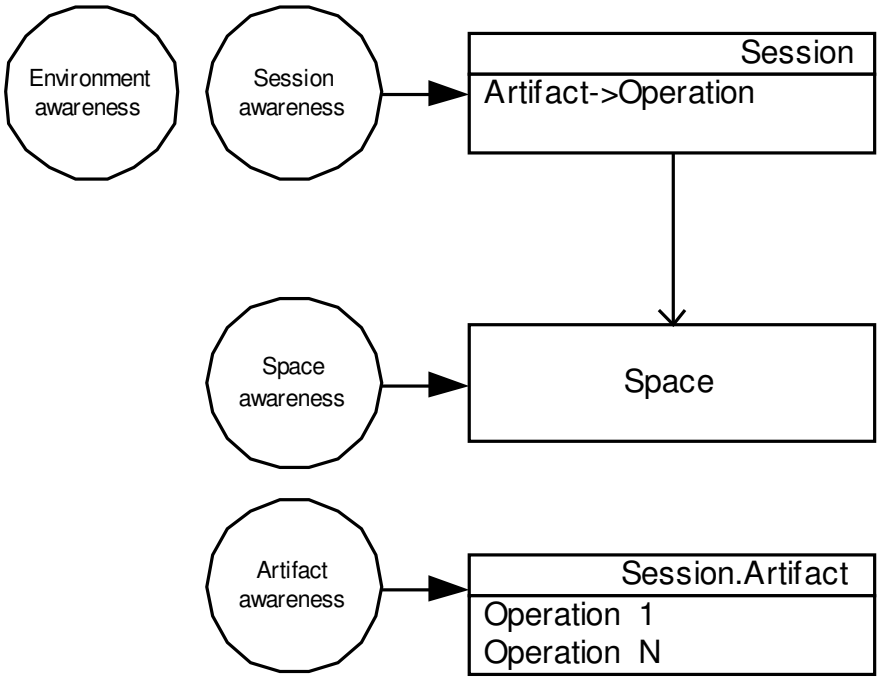


Ilustración 34: Notación de Awareness en los diferentes niveles

11. Conclusiones

Como hemos visto la construcción de sistemas colaborativos no es una tarea sencilla. En los últimos años han aparecido algunos frameworks [19], [20] que implementan alguna funcionalidad común como el manejo de objetos compartidos, los usuarios, los roles, etc. y que facilitan la etapa de construcción de este tipo de sistemas. Pero estos frameworks no alcanzan para incorporar estándares de calidad, soportar el reuso, facilitar el proceso de evolución y adaptación a las nuevas tecnologías. Esta desventaja principalmente se debe a que estas herramientas ponen foco en aspectos vinculados con la programación y no con el modelado. Debido a ello es que se vio la necesidad de contar con herramientas de modelado que facilitaran las etapas tempranas del desarrollo de los sistemas groupware. En esa línea dedicamos los primeros capítulos a repasar la historia de los sistemas colaborativos y a elaborar una lista de conceptos principales que fueron de gran utilidad a la hora del entendimiento de dichos sistemas.

En este contexto y principalmente por el hecho que MDD cambia el foco centrado en el código por el centrado en el modelo es que se notó que la aplicación de principios de MDD al desarrollo de sistemas colaborativos son una aproximación muy positiva para resolver los problemas mencionados. En este sentido se construyó un lenguaje específico de dominio (CSSL - Collaborative Software System Language) adaptado para expresar la semántica del dominio de los sistemas groupware. Los elementos del CSSL permiten a los desarrolladores crear una conceptualización abstracta de dominio de una manera simple, precisa y a la vez amigable. Además, los modelos escritos con CSSL son independientes del framework o herramientas con los que luego se van a construir. El CSSL fue construido como una extensión de UML usando el mecanismo de metamodelo y es soportado por herramientas open source que trabajan con UML y como veremos en los próximos pasos se implementó sobre la plataforma Eclipse para poder escribir los modelos

de los sistemas colaborativos. Luego también se verá más adelante como proceder a la transformación de esos modelos que permitirán llegar a la implementación del sistema concretamente.

En definitiva el CSSL nos provee una sintaxis abstracta. Uno de los beneficios de contar con una sintaxis abstracta es que podemos obtener distintas sintaxis concretas que se adapten a distintas necesidades. Por caso hemos descripto al menos dos sintaxis concretas - una similar al diseño de objetos y otra donde los modelos se obtienen completando formularios que están relacionados entre si - que permiten describir el sistema con mayor facilidad para usuarios con distintos niveles de conocimientos. Las sintaxis concretas se relacionan entre si y como veremos en el editor que se está desarrollando se pueden trabajar alternativamente.

Finalmente vimos que el lenguaje CSSL y las distintas sintaxis concretas obtenidas a partir de él requerían de una guía o método que ayudara a los diseñadores en la especificación de dichos sistemas. En el contexto de MDD era importante poner el foco en las etapas tempranas del desarrollo, ya que a partir de los principios de MDD los modelos subsiguientes se derivarían de los modelos previos. El método propuesto que está divido en etapas que se desarrollan en forma iterativa e incremental. Hace foco en aspectos de descripción del entorno colaborativo, basado en espacios de colaboración y protocolos de colaboración que se describirán en procesos de alto nivel donde cada paso se refinará en un protocolo más específico, llegando de definir que actividad puede desarrollar cada rol en un determinado momento.

11.1. Pasos a seguir

Como continuidad de este trabajo se están realizando actividades en el marco de dos trabajos de tesis que son de alguna manera productos que se obtuvieron a partir de este lenguaje de dominio presentado.

Por un lado tenemos un trabajo de tesis [8] en el que se estudiarán el uso de los frameworks de Eclipse para la definición de DSLs. En la actualidad existen numerosas extensiones que permiten definir e implementar lenguajes específicos, tanto como para la definición de su sintaxis abstracta, como para la concreta. En particular se implementará el lenguaje CSSL utilizando los frameworks ofrecidos por Eclipse para la definición de DSLs, como EMF, GMF, TMF.

Como resultado de este trabajo se espera obtener un plugin que permita a los diseñadores de sistemas colaborativos concebir las aplicaciones en un entrono que combina las dos sintaxis concretas que se plantearon. En el prototipo de plugin desarrollado se muestra en la Ilustración 35 la representación gráfica en el panel del medio, donde se relacionan los espacios, sesiones roles, objetos compartidos, etc. En el panel inferior la representación textual o de formularios. Los diseñadores pueden trabajar tanto en el panel gráfico como en la representación textual.

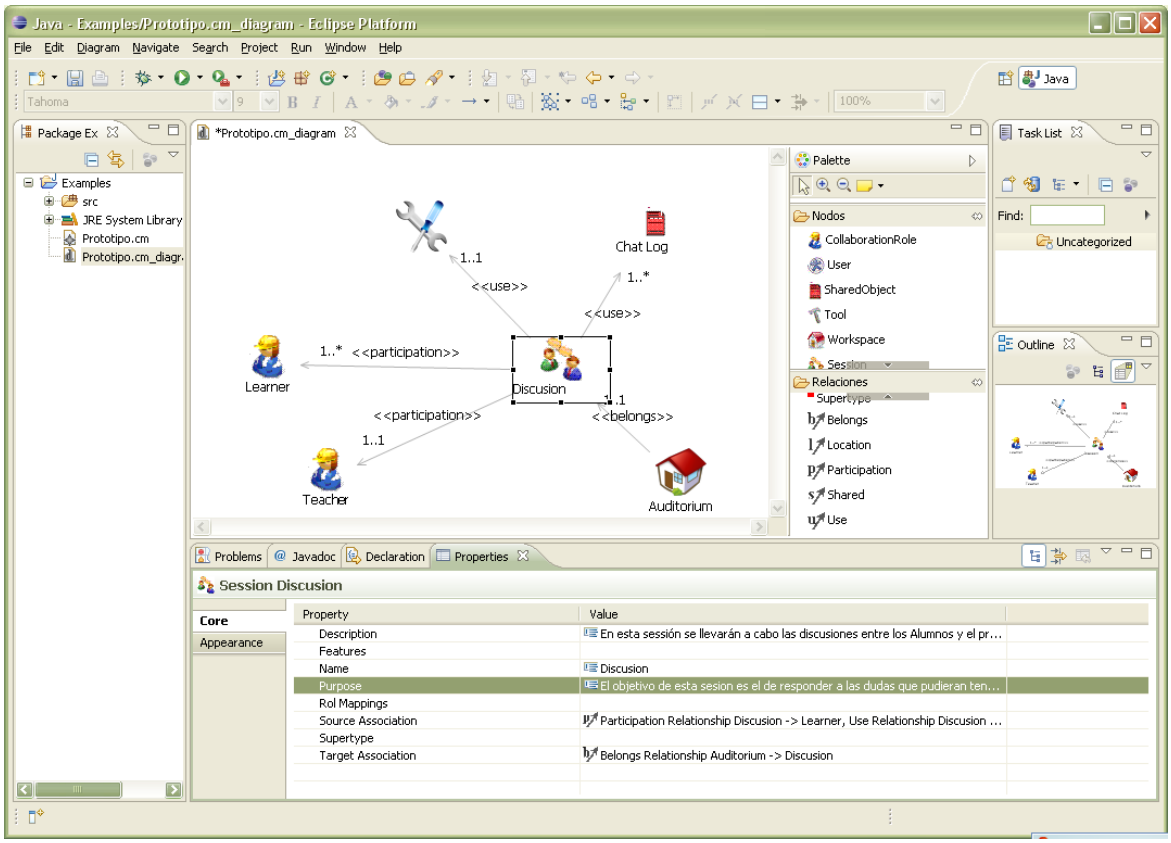


Ilustración 35: Layout del plugin para diseñar sistemas colaborativos

Por otro lado tenemos otro trabajo de tesis que tiene por objetivo la generación de código a partir de el metamodelo del lenguaje específico de dominio llamado “Collaborative Software System Language” (CSSL). Entre los mecanismos de generación estudiados, el elegido es simple, ya que las tecnologías usadas son muy similares a otras muy usadas (JSP, por ejemplo). Además, ya está siendo usado exitosamente en EMF, que está generando código desde hace varios años. Permitiría manipular y usar cualquier elemento expresado en EMF, y recorrer el modelo de manera libre.

Como resultado de la tesis se espera que el código generado tenga resuelto:

- Las estructuras estáticas de la aplicación (Los objetos que representan a usuarios, espacios, sesiones, objetos compartidos).

- Funcionalidad básica (login, etc…)

- La comunicación entre los clientes de algunos eventos.

- Transición entre estados de la sesión

No se espera que genere la aplicación completa, sino que tenga resuelta algunas situaciones generales de este tipo de sistemas y permita a los equipos de desarrollo agilizar en comienzo del desarrollo de este tipo de sistemas.

12. Referencias

1 - Ellis, C.A., Gibbs, S.J., Rein, G.L., Groupware: some issues and experiences, in: Communications of the ACM, 34(1) (1991).

2 - Grudin, Jonathan. "Computer-Supported Cooperative Work: History and Focus". Computer 27 (5): 19-26. IEEE. ISSN: 0018-9162. (1994).

3 - Bibbo, Luis Mariano; García, Diego; Pons, Claudia; "A Domain Specific Language for the Development of Collaborative Systems," *Chilean Computer Science Society, International Conference of the*, pp. 3-12, 2008 International Conference of the Chilean Computer Science Society, 2008.

4 - Bran Selic. The Pragmatics of Model- Driven Development. IEEE Software, 20(5), 19-25 (2003).

5 - Stahl, M Voelter. Model Driven Software Development. John Wiley, ISBN 0470025700. (2006).

6 - Object Management Group, MDA Guide, v1.0.1, omg/03-06-01 (2003).

7 - UML 2.0. The Unified Modeling Language Superstructure version 2.0 – OMG Final Adopted Specification, formal/2005-07-04. http://www.omg.org. (2005).

8 – Ruiz G., Andino L., Tesis de Licenciatura en Informática de la UNLP: "Análisis y uso de los frameworks de Eclipse para la definición de DSLs". Director: LM Bibbo. En curso. (2009).

9 - Mandel, Luis and Koch, Nora and Maier, Christoph. Extending UML to Model Hypermedia and Distributed Systems.Downloaded from http://projekte.fast.de/Projekte/forsoft/intoohdm/index.html.

10 - Rubart, Jessica and Dawabi, Peter. Shared data modeling with UML-G. International Journal of Computer Applications in Technology, Volume 19, Nos. 3/4, 2004.

11 - Dong, Ying and Li, Mingshu and Wang, Qing. A UML Extension of Distributed System. Proceedings of the Fist IEEE International Conference on Machine Learning and Cybernetics, Beijing, 4-5 November 2002.

12 - Bauer, B., Muller, J.P., and Odell, J. `Agent UML: a formalism for specifying multiagent interaction', Agent-Oriented Software Engineering, Springer-Verlag (held at ISCE '00, 2000), pp. 91-103. (2001).

13 - Pinheiro da Silva, P. and Paton, N.W. UMLi: the unified modeling language for interactive applications. Proceedings of the UML International Conference 2000, LNCS, Vol. 1939, pp. 117-132. (2000).

14 - Meta Object Facility (MOF) 2.0 Core Specification. OMG - (2005).

15 - Object Constraint Language OCL 2.0. OMG Final Adopted Specification. Document ptc/03-10-14. (2003).

16 - UML2. The Eclipse Model Development Tools (MDT) web site, http://www.eclipse.org/modeling/mdt/. It was accessed in May 2008.

17 - Guerrero Luis A. (1999) Design patterns for collaborative systems. Proceedings of the Fifth International Workshop on Groupware (CRIWG}

18 – Li Du and Muntz Richard R. (1999) A Collaboration Specification Language, In Proceedings of the 2nd USENIX Conference on Domain Specific Languages.

19 - Tietze, D.A. (2001) `A framework for developing compo-nent-based co-operative applications', GMD Research Series No. 7/2001, ISBN: 3-88457-390-X.

20 - Guicking, A., Tandler, P. and Avgeriou P. (2005) Agilo: A Highly Flexible Groupware Framework. In Book Groupware: Design, Implementation, and Use. Serie Lecture Notes in Computer Science, Springer Berlin / Heidelberg, Vol. 3706.

21 - Roseman, M. and S. Greenberg, 'Building real time groupware with GroupKit, a groupware toolkit'. ACM Transactions on Computer-Human Interaction, 3 (March 1996), 1, p. 66-106.

22 - Eclipse - an open development platform – http://www.eclipse.org.

23 - Query/View/Transformations (QVT) - OMG Adopted Specification. March 2005. http://www. omg.org.

24 - Jouault F. , Kurtev I. Transforming Models with ATL Workshop in Model Transformation in Practice at the MoDELS 2005 Conference. Montego Bay, Jamaica, Oct 3, 2005.

25 - MofScriptMOFScript Home page - www.eclipse.org/gmt/mofscript/.

26 - Schwabe Daniel nd Rossi Gustavo, ‘Developing hypermedia applications using OOHDM’, In Proceedings of Workshop on Hypermedia Development Process, Methods and Models, Hypertext´98; (1998) ACM.

27 – Henri Ter Hofte. Working Apart Together – Foundations for Component Groupware. Copyright © 1998, Telematica Instituut, The Netherlands ISBN 90-75176-14-7.

28 - Cockburn Andy and Jones Steve (1994) Four Principles for Groupware Design. Proceedings of OZCHI'94.

29 - Altenhofen, M., J. Dittrich, R. Hammerschmidt, T. Käppner, C. Kruschel, A. Kückes and T. Steining, 'The BERKOM multimedia collaboration service'. In Proceedings of ACM Multimedia 93, Anaheim, CA, USA, August 1-6, 1993. Association for Computing Machinery, New York, 1993, p. 457-463.

30 - Grudin, J., 'Why CSCW applications fail : Problems in the design and evaluation of organizational interfaces'. In [CSCW88], p. 85-93.

31 - Engelbart, D.C., 'NLS teleconferencing features : The journal, and shared-screen telephoning'. In CompCon75 Digest, September 9-11, 1975. IEEE, Los Alamitos, CA, USA, 1975, p. 173-176.

32 - Ellis, C.A. and S.J. Gibbs, 'Concurrency control in groupware systems'. In J. Clifford, B. Lindsday and D. Maier (eds.), Proceedings of the ACM SIGMOD'89 conference on the management of data, Seattle, WA, USA, May 2-4, 1989. ACM Press, New York, 1989, p. 399-407.

33 - Benford, S.D., H. Smith, A. Sheperd, A. Bullock and H. Howidy, 'Information sharing approach to CSCW : The Grace project'. Computer communications, 15 (1992), 8, p. 502-509.

34 - Applegate, L.M., B.R. Konsyns ki and J.F. Nunamaker, 'A group decision support system for idea generation and issue analysis in organization planning'. In [CSCW86], p. 16-34.

35 - Kraemer, K.L. and J.L. King, 'Computer-based systems for cooperative work and group decision making : Status of use and problems in development'. In [CSCW86], p. 353-375.

36 - Kranz, M.E. and V.I. Sessa, 'Meeting makeovers : Electronic meeting support'. PC Magazine, 13 (June 14, 1994), 11, p. 205-207, 210-212.

37 - Leland, M.D.P., R.S. Fish and R.E. Kraut, 'Collaborative document production using Quilt'. In [CSCW88], p. 206-215.

38 - Ellis, C.A., S.J. Gibbs and G.L. Rein, 'Design and use of a group editor'. In G. Cockton (ed.), Engineering for human-computer interaction : proceedings of the IFIP TC2/WG2.7 working conference on engineering for human-computer interaction, Napa-Valley, CA, USA, 21-25 August 1989. North-Holland, Amsterdam, 1990, p. 13-28.

39 - Greenberg S. and Roseman Mark (1998), Using a room metaphor to ease transitions in groupware.

40 - Fish, R.S., R.E. Kraut, M.D.P. Leland and M. Cohen, 'Quilt : A collaborative tool for cooperative writing'. In R.B. Allen (ed.), Conference on office information systems, March 23-25, 1988, Palo Alto, CA, USA, vol. 9(2&3), SIGOIS bulletin. ACM Press, New York, 1988, p. 30-37.

41 - Greenberg, S., M. Roseman, D. Webster and R. Bohnet, 'Human and technical factors of distributed group drawing tools'. Interacting with computers, 4 (1992), 1, p. 364-392.

42 - Wilson, B., 'Wscrawl 2.0: a shared whiteboard based on X-windows'. In S. G reenberg, S. Hayne and R. Rada (eds.), Groupware for real time drawing: A designer's guide. McGraw-Hill, New York, 1995, p. 129-141.

43 - Haake, J.M. and B. Wilson, 'Supporting collaborative writing of hyperdocuments in SEPIA'. In [CSCW92], p. 138-146,

44 - Stefik, M., G. Foster, D.G. Bobrow, K. Kahn, S. Lanning and L. Suchman, 'Beyond the chalkboard : Computer support for collaboration and problem solving in meetings'. Communications of the ACM, 30 (January 1987), 1, p. 32-47.

45 - Labriola, D., 'Remote possibilities : Whiteboard software'. PC Magazine, 13 (June 14, 1994), 11, p. 223-228.

46 - Nunamaker, J.F., A.R. Dennis, J.S. Valacich, D.R. Vogel and J.F. George, 'Electronic meeting systems to support group work'. Communications of the ACM, 34 (July 1991), 7, p. 40-61.

47 - Takemura, H. and F. Kishino, 'Cooperative work environment using virtual workspace'. In [CSCW92], p. 226-232.

48 - Grundy, J.C., J.G. Hosking and W.B. Mugridge, 'Low-level and high-level CSCW support in the serendipity process modelling environment'. In J.C. Grundy and M.D. Apperley (eds.), OzCHI'96 : Proceedings of Sixth Australian conference on computer-human interaction, Hamilton, New Zealand, November 24-27, 1996. IEEE Computer Society Press, Los Alamitos, CA, USA, 1996, p. 69-76.

49 - Brothers, L., V. Sembugamoorthy and M. Muller, 'ICICLE: Groupware for code inspection'. In [CSCW90], p. 169-181.

50 - Churcher, N. and C. Cerecke, 'GroupCRC: Exploring CSCW support for software engineering'. In J.C. Grundy and M.D. Apperley (eds.), OzCHI'96 : Proceedings of Sixth Australian conference on computer-human interaction, Hamilton, New Zealand, November 24-27, 1996. IEEE Computer Society Press, Los Alamitos, CA, USA, 1996, p. 62-68.

51 - Bandinelli, S., E. Di Nitto and A. Fugetta, 'Supporting cooperation in the SPADE-1 environment'. IEEE Transactions on software engineering, 22 (December 1996), 12, p. 841-885.

52 - Danielsen, T., U. Pankoke-Babatz, W. Prinz, A. Patel, P.A. Pays, K. Smaaland and R. Speth, 'The AMIGO project : Advanced group communication model for computer-based communication environment'. In [CSCW86], p. 115-142.

53 - Hill, R.D., T. Brinck, S.L. Rohall, J.F. Patterson and W. Wilner, 'the rendezvous architecture and language for constructing multiuser applications'. ACM Transactions on Computer-Human Interaction, 1 (June 1994), 2, p. 81-125.

54 - Victor, F. and E. Sommer, 'Supporting the design of office procedures in the DOMINO system'. In J.M. Bowers and S.D. Benford (eds.), Studies in computer supported cooperative work : theory, practice and design, vol. 8, Human factors in information technology. North-Holland, Amsterdam, 1991, p. 119-125.

55 - Riexinger, D. and K. Werner, 'Integration of existing applications into a conference system'. In R. Steinmetz (ed.), Multimedia : Advanced teleservices and high-speed communication architectures : Proceedings of Second international workshop, IWACA'94, Heidelberg, Germany, September 26-28, 1994, vol. 868, Lecture notes in computer science. Springer-Verlag, Berlin, 1994, p. 346-355.

56 - LaLiberte, D. and A. Braverman, A protocol for scalable group and public annotations, 1995. NCSA.

57 - Lauwers, J.C. and K.A. Lantz, 'Collaboration awareness in support of collaboratio n transparency: Requirements for the next generation of shared window systems'. In J.C. Chew and J. Whiteside (eds.), Human factors in computing systems : CHI '90, "Empowering people", conference, SIGCHI bulletin. ACM Press, New York, 1990, p. 303-311.

58 - Gutwin, C. and Greenberg, S. (2002) A Descriptive Framework of Workspace Awareness for Real-Time Groupware. Computer Supported Cooperative Work (CSCW), The Journal of Collaborative Computing, Kluwer Academic Press.

71 - Bowers, J.M., J. Churcher and T. Roberts, 'Structuring computer-mediated communication in COSMOS'. In R. Speth (ed.), Research into networks and distributed applications : Proceedings of European teleinformatics conference - EUTECO '88 on research into networks and distributed applications, Vienna, Austria, April 20-22, 1988. North-Holland, Amsterdam, 1988, p. 195-209.

72 - Garfinkel, D., B.C. Welti and T.W. Yip, 'HP SharedX : A tool for real-time collaboration'. Hewlett-Packard Journal, 45 (April 1994), 2, p. 23-36.